

Symmetry, Outer Bounds, and Code Constructions: A Computer-Aided Investigation on the Fundamental Limits of Caching

Chao Tian *

December 14, 2016

Abstract

We considered the optimal memory-transmission-rate tradeoff of caching systems. Different from the conventional analytical approach usually seen in the information theory literature, we rely on a computer-aided approach in this investigation. The linear programming (LP) outer bound of the entropy space serves as the starting point of this approach, however our effort goes significantly beyond using it to prove information inequalities. We first identify and formalize the symmetry structure in the problem, which enables us to show the existence of optimal symmetric solutions. A symmetry-reduced linear program is then used to identify the boundary of the memory-transmission-rate tradeoff for several simple cases, for which we obtain a set of tight outer bounds. General hypotheses on the optimal tradeoff region are formed from these computed data, which are then analytically proved. This leads to a complete characterization of the optimal tradeoff for systems with only two users, and certain partial characterization for systems with only two files. Next, we show that by carefully analyzing the joint entropy structure of the outer bounds for certain cases, a novel code construction can be reverse-engineered, which eventually leads to a general class of codes. Finally, we show that strong outer bounds can be computed through strategically relaxing the LP. This allows us firstly to deduce generic characteristic of the converse proof, and secondly to compute outer bounds for larger problem cases, despite the seemingly impossible computation scale.

Index terms— Computer-aided analysis, information theory.

1 Introduction

Caching is a data management technique that can alleviate the communication burden during peak traffic time, by prefetching and prestoring certain useful content at the users' local caches during off-peak hours. Maddah-Ali and Niesen [1] recently considered the problem in an information theoretical framework. In this setting, there are a total of N mutually independent files of equal size as well as K users. The overall system operates in two phases: in the placement phase, each user stores in his local cache some content from these files; in the delivery phase, each user will request one file, and the central server transmits (multicasts) certain common content to all the users to accommodate their requests. Each user has a local cache memory of capacity M , and the contents stored in the placement phase are determined without knowing a priori the precise requests in the delivery phase. The system should minimize the amount of multicast information which has rate R for all possible combinations of user requests, under the memory cache constraint M , both of which are measured as multiples of the file size. It is clear that there is a tradeoff between R and M , and thus the fundamental question is the characterization of the optimal tradeoff.

*C. Tian is with the Department of Electrical Engineering and Computer Science, The University of Tennessee, Knoxville, TN 37996, USA (email: chao.tian@utk.edu). This paper was presented in part at 2016 Intentional Symposium on Information Theory, Barcelona Spain, July 2016. The work of C. Tian was supported in part by the National Science Foundation under Grant CCF-15-26095.

It was shown in [1] that coding can be very beneficial in this setting, and in fact provides a tradeoff within a constant factor of the optimum, while uncoded solutions suffer a significant loss. Subsequent works extended it to decentralized caching placements [2], caching with nonuniform demands [3], and online caching placements [4], among other things. In many practical situations, the possible gain by using optimal codes instead of the suboptimal ones can still be significant, and both the outer bounds and inner bounds are believed to be loose in general. As such, identifying more precisely the fundamental limits of caching systems are of critical importance. Indeed, there are significant research activities recently [5–12] in both refining the outer bounds and finding stronger codes.

Despite these efforts, the fundamental tradeoff between the cache memory capacity M and the content delivery rate R has not been fully characterized except for the case when $(N, K) = (2, 2)$, which was given in [1]. This is partly due to the fact that the main focus of the initial investigations [1–4] was on systems operating in the regime where the number of files N and the number of users K are both large, for which the coded solutions can provide the largest gain over the uncoded counterpart. However, in many applications, the number of simultaneous data requests can be small, or the collection of users or files need to be divided into subgroups in order to account for various service and request inhomogeneities [3]. More importantly, precise and conclusive results on such cases with small N or K usually provide insights into more general cases, as we shall show in this work.

In this work, we investigate the fundamental limits of the caching systems. In contrast to the conventional analytical approach usually seen in the information theory literature, our investigation has a strong computer-aided flavor. Using the linear programming (LP) outer bound of the entropy space [13] as the starting point, we develop methods to compute outer bounds in caching systems, and more importantly, show that data obtained through computation can be used in several different manners to deduce meaningful structural understanding of the fundamental limits.

In order to utilize the computational tool, the symmetry structure in the problem must be understood and then used to reduce the problem to a manageable scale. The symmetry-reduced LP is then used to identify the boundary of the memory-transmission-rate tradeoff for several simple cases. General hypotheses on the optimal tradeoff region are formed from this data, which are then analytically proved. This leads to a complete characterization of the optimal tradeoff for systems with only two users, and certain partial characterization for systems with only two files. Next, we show that by carefully analyzing the joint entropy structure of the outer bounds for certain cases, a novel code construction can be reverse-engineered, which eventually leads to a general class of codes. Moreover, data can also be used to show certain tradeoff pair is not achievable using linear codes. Finally, we show that strong outer bounds can be computed through strategically relaxing the LP. This allows us firstly to deduce generic characteristic of the converse proof, and secondly to compute outer bounds for larger problem cases, despite the seemingly impossible computation scale.

Although some of the best and the most conclusive results on the optimal memory-transmission-rate tradeoff in caching systems are presented in this work, one focus of this work is to provide several generic computer-aided methods that can be used to facilitate such investigations. We believe these methods are quite general, and can be applied to other coding and communication problems, particularly those related to data storage and data management. It is our belief that this work provides strong evidence that more machine intelligence can be effectively used in the field of information theory and communication research, particularly on the investigation of fundamental limits. Indeed, our effort can be viewed as both data-driven and computational, and thus more advanced data analysis and machine learning technique may prove useful.

The rest of the paper is organized as follows. In Section 2, existing results on the caching problem and some background information on the entropy LP framework are reviewed. The symmetry structure of the caching problem is identified and then used to show the existence of optimal symmetric solutions in Section 3. In Section 4 we show how the data obtained through computation can be used to form data-driven hypotheses, and then analytically prove these hypotheses. In Section 5 we show that the computed data can also be used to facilitate reverse-engineering new codes, and also to prove that certain memory-

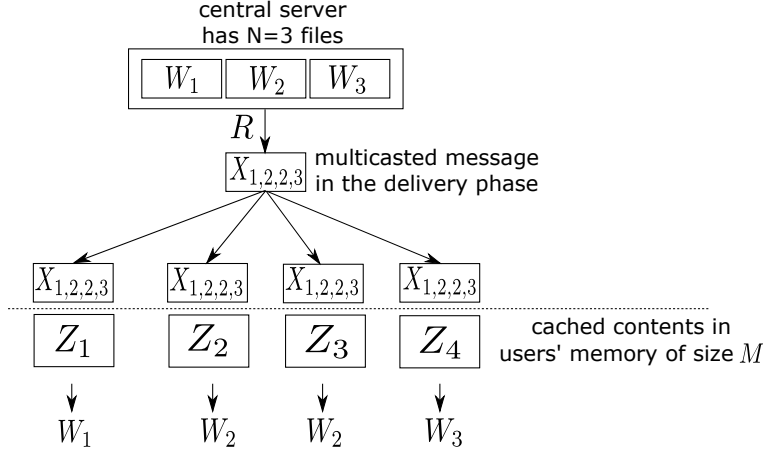


Figure 1: An example caching system instance, where there are $N = 3$ files and $K = 4$ users. In this instance the users request files $(1, 2, 2, 3)$, respectively, and thus the multicast common information is written as $X_{1,2,3,4}$.

transmission-rate pair is not achievable using linear codes. In Section 6, we explore the structure of the outer bounds computationally, and show that strong outer bounds can be obtained by enforcing only certain subsets of the constraints. This observation then enables us to produce strong bounds for larger problem cases, and some of these results thus obtained are presented and discussed. A few concluding remarks are given in Section 7. Several technical proofs are relegated to the appendix.

2 Preliminaries

2.1 The Caching System Model

There are a total of N mutually independent files of equal size and K users in the system. As mentioned earlier, the overall system operates in two phases: the placement phase and the delivery phase. In the first phase, each user stores in his local cache some content, which has a capacity M , without the knowledge of the requests in the second phase; in the second phase, each user requests one file, and the same central server transmits (multicasts) certain common content, which has rate R , to all the users to accommodate their requests. The system should minimize R under the constraint M , both of which are measured as multiples of the file size. The primary interest of this work is the optimal tradeoff between M and R . In the rest of the paper, we shall refer to a specific combination of the file requests of all users together as a *demand*, or a *demand pattern*, and reserve the word “request” as the particular file a user needs.

Since we are investigating the fundamental limits of the caching systems in this work, the notation for the various quantities in the systems needs to be specified. The N files in the system are denoted as $\mathcal{W} \triangleq \{W_1, W_2, \dots, W_N\}$, the cached contents at the K users are denoted as $\mathcal{Z} \triangleq \{Z_1, Z_2, \dots, Z_K\}$, and the transmissions to satisfy a given demand is denoted as X_{d_1, d_2, \dots, d_K} , *i.e.*, the transmitted information X_{d_1, d_2, \dots, d_K} when user k requests file W_{d_k} , $k = 1, 2, \dots, K$. Occasionally, we shall write (W_1, W_2, \dots, W_n) simply as $W_{[1:n]}$, and similarly $(X_{k,1}, X_{k,2}, \dots, X_{k,n})$ as $X_{k,[1:n]}$, and (d_1, d_2, \dots, d_K) as $d_{[1:K]}$. There are other simplifications of the notation for certain special cases of the problem, which will be introduced as they become necessary.

The cache contents are produced directly from the files, and the encoding function to produce the cached content at the k -th user is denoted as $Z_k = f_k(W_{[1:N]})$; similarly the transmission contents are also produced directly from the files, and the encoding functions are denoted as $X_{d_{[1:K]}} = g_{d_{[1:K]}}(W_{[1:N]})$, which depends on the particular demands $d_{[1:K]}$. Since the cached contents and transmitted information

are both deterministic functions of the files, we have

$$H(Z_k|W_1, W_2, \dots, W_N) = 0, \quad k = 1, 2, \dots, K, \quad (1)$$

$$H(X_{d_1, d_2, \dots, d_K}|W_1, W_2, \dots, W_N) = 0, \quad d_k \in \{1, 2, \dots, N\}. \quad (2)$$

It is also clear that

$$H(W_{d_k}|Z_k, X_{d_1, d_2, \dots, d_K}) = 0, \quad (3)$$

i.e., the file W_{d_k} is a function of the cached content Z_k at user k and the transmitted information when user k requests W_{d_k} . The memory satisfies the constraint

$$M \geq H(Z_i), \quad i \in \{1, 2, \dots, K\}, \quad (4)$$

and the transmission rate satisfies

$$R \geq H(X_{d_1, d_2, \dots, d_K}), \quad d_k \in \{1, 2, \dots, N\}. \quad (5)$$

Any valid caching code must satisfy the specific set of conditions in (2)-(5).

2.2 Known Results on Caching Systems

The first achievability result on this problem was given in [1], which is directly quoted below.

Theorem 2.1 (Maddah-Ali and Niesen [1]). *For N files and K users each with a cache of size $M \in \{0, N/K, 2N/K, \dots, N\}$,*

$$R = K(1 - M/N) \cdot \min \left\{ \frac{1}{1 + KM/N}, \frac{N}{K} \right\} \quad (6)$$

is achievable. For general $0 \leq M \leq N$, the lower convex envelope of these (M, R) points is achievable.

The first term in the minimization is achieved by the scheme of uncoded placement together with coded transmission [1], while the latter term is by simple uncoded placement and uncoded transmission. In a recent work [8], Chen *et al.* extended a special scheme for the case $N = K = 2$ discussed in [1] to the general case $N \leq K$, and showed that the tradeoff pair $\left(\frac{1}{K}, \frac{N(K-1)}{K}\right)$ is achievable, which is in fact optimal. The scheme given in [1] uses uncoded placement with coded transmission, while the scheme in [8] uses coded placement and coded transmission. There were several additional efforts in attempting to find better codes [9–11], however, it appears that although these codes can provide improved memory-transmission-rate tradeoff in some cases, none of them gives any additional points on the optimal tradeoff. Independent of these works, Tian and Chen [12] proposed a class of new codes for $N \leq K$, the origin of which will be discussed in more details in Section 5; several points achieved by this new class of codes are indeed optimal. It should also be noted that while all the previous schemes [1, 8–10] essentially use only binary codes, the codes in [12] use a more general finite field.

A cut-set outer bound was also given in [1], which is again directly quoted below.

Theorem 2.2 (Maddah-Ali and Niesen [1]). *For N files and K users each with a cache of size $0 \leq M \leq N$,*

$$R \geq \max_{s \in \{1, 2, \dots, \min\{N, K\}\}} \left(s - \frac{sM}{\lfloor N/s \rfloor} \right). \quad (7)$$

Several efforts to improve this outer bound have also been reported [5–7]. However, as mentioned earlier, even for the simplest cases beyond $(N, K) = (2, 2)$, complete characterizations are still elusive. In this work we specifically treat such small problem cases, and attempt to deduce more generic properties and outer bounds from these cases. Whenever possible and meaningful, we shall compare our bounds with existing results in the literature, either analytically or numerically.

2.3 The Basic Linear Programming Framework

The basic linear programming bound on the entropy space was introduced by Yeung [13], which can be understood as follows. Consider a total of n discrete random variables (X_1, X_2, \dots, X_n) with a given joint distribution. There are a total of $2^n - 1$ joint entropies, each associated with a non-empty subset of these random variables. It is known that the entropy function is monotone and submodular, and thus any valid $(2^n - 1)$ dimensional entropy vector must have such properties, which can be written as a set of inequalities. Yeung showed (see *e.g.*, [14]) that the minimal sufficient set of such inequalities is the so-call elemental inequalities

$$H(X_i | \{X_k, k \neq i\}) \geq 0, \quad i \in \{1, 2, \dots, n\} \quad (8)$$

$$I(X_i; X_j | \{X_k, k \in \Phi\}) \geq 0, \text{ where } \Phi \subseteq \{1, 2, \dots, n\} \setminus \{i, j\}, i \neq j. \quad (9)$$

The $2^n - 1$ joint entropy terms can be viewed as the variables in a linear programming (LP) problem, and there are total of $n + \binom{n}{2} 2^{n-2}$ constraints in (8)-(9). In addition to this generic set of constraints, each specific coding problem will place additional constraints on the joint entropy values. These can be viewed as a constraint set of the given problem, although the problem might also induce constraints that are not in this form or even not possible to write in terms of joint entropies. For example, in the caching problem, the set of random variables are $\{W_i, i = 1, 2, \dots, N\} \cup \{Z_i, i = 1, 2, \dots, K\} \cup \{X_{d_1, d_2, \dots, d_K} : d_k \in \{1, 2, \dots, N\}\}$, and there are a total of $2^{N+K+N^K} - 1$ variables in this LP, the problem specific constraints are those in (2)-(5), and there are $N + K + N^K + \binom{N+K+N^K}{2} 2^{N+K+N^K-2}$ elemental entropy constraints, which is in fact doubly exponential in the number of users K .

2.4 A Computed Aided Approach to Find Outer Bounds

In principle, with the afore-described constraint set, one can simply convert the outer bounding problem into an LP, and use a generic LP solver to compute it. Unfortunately, despite the effectiveness of modern LP solvers, directly applying this approach on an engineering problem is usually not possible, since the scale of the LP is often very large even for simple settings. For example, for the caching problem, when $N = 2, K = 4$, there are overall 200 million elemental inequalities. The key insight used in [15] to make the problem tractable is that the LP can usually be significantly reduced, by taking into account of the symmetry and the implication relations in the problem.

The details of the reductions can be found in [15], and here we only provide two examples in the context of the caching problem to illustrate the basic idea behind these reductions:

- Assuming the optimal codes are symmetric, which will be defined more precisely later, the joint entropy $H(W_2, Z_3, X_{2,3,3})$ should be equal to the joint entropy $H(W_1, Z_2, X_{1,2,2})$. This implies that in the LP, we can represent both quantities using a single variable.
- Because of the relation (3), the joint entropy $H(W_2, Z_3, X_{2,3,3})$ should be equal to the joint entropy $H(W_2, W_3, Z_3, X_{2,3,3})$. This again implies that in the LP, we can represent both quantities using a single variable.

The reduced primal LP problem is usually significantly smaller, which allows us to find a lower bound for the tradeoff region for a specific instance with fixed file sizes. Moreover, after identifying the region of interest using these computed boundary points¹, a human-readable proof can also be produced

¹In [15], the region of interest was obtained by first finding a set of fine-spaced points on the boundary of the outer bound using the reduced LP, and then manually identifying the effective bounding segments using these boundary points. This task can however be accomplished more efficiently using an approach proposed by Lassez and Lassez [16]. The author wishes to acknowledge the discussion with John M. Walsh and Jayant Apte at Drexel University regarding this issue (see also [17]), which prompted him to implement this part of the computer program using this more efficient approach. For completeness, the specialization of the Lassez algorithm to the caching problem, which is much simplified in this setting, is provided in the Appendix.

computationally by invoking the dual of the LP given above. More details can again be found in [15], however, this procedure can be intuitively viewed as follows. Suppose a valid outer bound in the constraint set has the form of

$$\sum_{\Phi \subseteq \{1,2,\dots,n\}} \alpha_{\Phi} H(X_k, k \in \Phi) \geq 0, \quad (10)$$

then it must be a linear combination of the known inequalities, *i.e.*, (8)-(9), and the problem specific constraints, *e.g.*, (2)-(5) for the caching problem. To find a human readable proof is essentially to find a valid linear combination of these inequalities, and for the conciseness of the proof, the sparsest linear combination is preferred. By utilizing the LP dual with an additional linear objective, we can find within all valid combinations a sparse (but not necessarily the sparsest) one, which can yield a concise proof of the inequality (10).

The proof found through this approach can be conveniently written as a table, or more precisely, a matrix to list all the linear combination coefficients, and one can easily produce a chain of inequalities using such a table to obtain a more conventional human-readable proof. This approach of generating human-readable proofs has subsequently been adopted by other researchers [18, 19]. Though we shall present several results thus obtained in this current work in the tabulation form, our main goal here is not to provide the most comprehensive bounds, but to use these results to present further developments of the computer-aided approach beyond simply proving bounds for small problem cases, and show the effectiveness of our approach.

3 Symmetry in the Caching Problem

The computer-aided approach to derive outer bounds mentioned earlier relies critically on the reduction of the basic entropy LP using symmetry and other problem structure. In this section, we consider the symmetry in the caching problem. Before presenting the symmetry structure formally, let us consider it in a more intuitive manner. Suppose a code for placement and transmission is given. If we place the cached contents in a permuted manner at the users, it will lead to a new code that is equivalent to the original one. Similarly, if we reorder the files and apply the same encoding function, the transmissions can also be changed accordingly to accommodate the requests, which is again an equivalent code. The two types of symmetries can be combined, and they induce a permutation group on the joint entropies of the subsets of the random variables $\mathcal{W} \cup \mathcal{Z} \cup \mathcal{X}$.

For concreteness, we sometimes specialize to the case $(N, K) = (3, 4)$ in the discussion below, and for this case

$$\begin{aligned} \mathcal{W} &= \{W_1, W_2, W_3\}, \quad \mathcal{Z} = \{Z_1, Z_2, Z_3, Z_4\}, \\ \mathcal{X} &= \{X_{d_1, d_2, d_3, d_4} : d_k \in \{1, 2, 3\}\}. \end{aligned} \quad (11)$$

3.1 Symmetry in User Indexing

Let a permutation function be defined as $\bar{\pi}(\cdot)$ on the user index set of $\{1, 2, \dots, K\}$, which reflects the permuted placement of cached contents \mathcal{Z} , *i.e.*, $\bar{\pi}(Z_k) = Z_{\bar{\pi}(k)}$. Let the inverse of $\bar{\pi}(\cdot)$ be denoted as $\bar{\pi}^{-1}(\cdot)$, then $\bar{\pi}(\cdot)$ also induces a permutation on \mathcal{X} as

$$\bar{\pi}_x(X_{d_1, d_2, \dots, d_K}) = X_{d_{\bar{\pi}^{-1}(1)}, d_{\bar{\pi}^{-1}(2)}, \dots, d_{\bar{\pi}^{-1}(K)}}. \quad (12)$$

For example, the permutation function $\bar{\pi}(1) = 2, \bar{\pi}(2) = 3, \bar{\pi}(3) = 1, \bar{\pi}(4) = 4$ maps Z_1 to $\bar{\pi}(Z_1) = Z_2$, but maps $X_{1,2,3,2}$ to $X_{3,1,2,2}$, $X_{3,2,1,3}$ to $X_{1,3,2,3}$, and $X_{1,1,2,2}$ to $X_{2,1,1,2}$. We define the mapping on a collection of such random variables as the collection of the elements after mapping each element individually.

We call a caching code user-index-symmetric, if for any subsets $\mathcal{W}_o \subseteq \mathcal{W}, \mathcal{Z}_o \subseteq \mathcal{Z}, \mathcal{X}_o \subseteq \mathcal{X}$, and any permutation $\bar{\pi}$, the following relation holds

$$H(\mathcal{W}_o, \mathcal{Z}_o, \mathcal{X}_o) = H(\mathcal{W}_o, \bar{\pi}(\mathcal{Z}_o), \bar{\pi}_x(\mathcal{X}_o)). \quad (13)$$

For example, for such a symmetric code, the entropy $H(W_2, Z_2, X_{1,2,3,2})$ under the aforementioned permutation is equal to $H(W_2, Z_3, X_{3,1,2,2})$; note that W_2 is a function of $(Z_2, X_{1,2,3,2})$, and after the mapping it is a function of $(Z_3, X_{3,1,2,2})$.

From a coding perspective, we can define the permutation through a set of new encoding functions. Given the original encoding functions f_k and $g_{d[1:K]}$ which will induce a probability distribution among $\mathcal{W} \cup \mathcal{Z} \cup \mathcal{X}$ and consequently $2^{N+K+N^K} - 1$ joint entropy values, the new functions $f_k^{\bar{\pi}}$ and $g_{d[1:K]}^{\bar{\pi}}$ corresponding to a permutation $\bar{\pi}$ are:

$$\begin{aligned} f_k^{\bar{\pi}}(W_{[1:N]}) &= f_{\bar{\pi}(k)}(W_{[1:N]}), \\ g_{d[1:K]}^{\bar{\pi}}(W_{[1:N]}) &= g_{d_{\bar{\pi}^{-1}([1:K])}}(W_{[1:N]}). \end{aligned} \quad (14)$$

3.2 Symmetry in File Indexing

Let a permutation function be defined as $\hat{\pi}(\cdot)$ on the file index set of $\{1, 2, \dots, N\}$, which reflects the renaming of the files \mathcal{W} , *i.e.*, $\hat{\pi}(W_k) = W_{\hat{\pi}(k)}$. It induces a permutation on \mathcal{X}

$$\hat{\pi}_x(X_{d_1, d_2, \dots, d_K}) = X_{\hat{\pi}(d_1), \hat{\pi}(d_2), \dots, \hat{\pi}(d_K)}. \quad (15)$$

For example, the permutation function $\hat{\pi}(1) = 2, \hat{\pi}(2) = 3, \hat{\pi}(3) = 1$ maps W_2 to $\hat{\pi}(W_2) = W_3$, but maps $X_{1,2,3,2}$ to $X_{2,3,1,3}$, $X_{3,2,1,3}$ to $X_{1,3,2,1}$, and $X_{1,1,2,2}$ to $X_{2,2,3,3}$.

We call a caching code file-index-symmetric, if for any subsets $\mathcal{W}_o \subseteq \mathcal{W}, \mathcal{Z}_o \subseteq \mathcal{Z}, \mathcal{X}_o \subseteq \mathcal{X}$, and any permutation $\hat{\pi}$, the following relation holds

$$H(\mathcal{W}_o, \mathcal{Z}_o, \mathcal{X}_o) = H(\hat{\pi}(\mathcal{W}_o), \mathcal{Z}_o, \hat{\pi}_x(\mathcal{X}_o)). \quad (16)$$

For example, for such a symmetric code, $H(W_3, Z_3, X_{1,2,3,2})$ under the aforementioned permutation is equal to $H(W_1, Z_3, X_{2,3,1,3})$; note that W_3 is a function of $(Z_3, X_{1,2,3,2})$, and after the mapping W_1 is a function of $(Z_3, X_{2,3,1,3})$.

From a coding perspective, we can again define the permutation through a set of new encoding functions. Given original encoding functions f_k and $g_{d[1:K]}$, the new functions $f_k^{\hat{\pi}}$ and $g_{d[1:K]}^{\hat{\pi}}$ associated with $\hat{\pi}$ are:

$$\begin{aligned} f_k^{\hat{\pi}}(W_{[1:N]}) &= f_k(W_{\hat{\pi}([1:N])}), \\ g_{d[1:K]}^{\hat{\pi}}(W_{[1:N]}) &= g_{\hat{\pi}(d[1:K])}(W_{\hat{\pi}([1:N])}). \end{aligned} \quad (17)$$

3.3 Existence of Optimal Symmetric Codes

With the symmetry structure elucidated above, we can now state our first auxiliary result.

Proposition 3.1. *For any caching code, there is a code with the same or smaller caching memory and transmission rate, which is both user-index-symmetric and file-index-symmetric.*

We call a code that is both user-index-symmetric and file-index-symmetric a *symmetric code*. This proposition implies that there is no loss of generality to consider only symmetric codes. The proof of this proposition relies on a simple space-sharing argument where each file is partitioned into a total of $N!K!$ segments, and the i -th set of permuted (both in user-indexing and file-indexing) encoding functions are used on the i -th segments of all the files to produce random variables $\mathcal{W}^{(i)} \cup \mathcal{Z}^{(i)} \cup \mathcal{X}^{(i)}$, $i = 1, 2, \dots, N!K!$.

Table 1: Demand types for small (N, K) pairs

(N, K)	Demand types
(2, 3)	(3, 0), (2, 1)
(2, 4)	(4, 0), (3, 1), (2, 2)
(3, 2)	(2, 0, 0), (1, 1, 0)
(3, 3)	(3, 0, 0), (2, 1, 0), (1, 1, 1)
(3, 4)	(4, 0, 0), (3, 1, 0), (2, 2, 0), (2, 1, 1)
(4, 2)	(2, 0, 0, 0), (1, 1, 0, 0)
(4, 3)	(3, 0, 0, 0), (2, 1, 0, 0), (1, 1, 1, 0)

It is not difficult to see that for any joint entropy $H(\mathcal{W}_o^{(1)} \cup \mathcal{Z}_o^{(1)} \cup \mathcal{X}_o^{(1)})$, where $\mathcal{W}_o^{(1)} \subseteq \mathcal{W}^{(i)}$, $\mathcal{Z}_o^{(1)} \subseteq \mathcal{Z}^{(1)}$ and $\mathcal{Z}_o^{(1)} \subseteq \mathcal{Z}^{(1)}$, the i -th permutation through composition $\bar{\pi} \cdot \hat{\pi}$ will produce a joint entropy $H(\hat{\pi}(\mathcal{W}_o^{(i)}) \cup \bar{\pi}(\mathcal{Z}_o^{(i)}) \cup \bar{\pi} \cdot \hat{\pi}(\mathcal{X}_o^{(i)}))$ of the same value. It is now clear that the resultant code by space sharing is indeed symmetric, and it has (normalized) memory sizes and transmission rate no worse than the original one. In a separate work, we investigate the properties of the induced permutation $\bar{\pi} \cdot \hat{\pi}$ in more details, and particularly, showed that the composition is commutative; readers are referred to [20] for more details.

3.4 Demand Types

Even for symmetric codes, the transmissions to satisfy different types of demands may use different rates. For example in the setting $N, K = (3, 4)$, $H(X_{1,2,2,2})$ may not be equal to $H(X_{1,1,2,2})$, and $H(X_{1,2,3,2})$ may not be equal to $H(X_{3,2,3,2})$. The transmission rate R is then chosen to be the maximum among all cases. This motivates the notion of demand types.

Definition 3.2. *In an (N, K) caching system, for a specific demand, let the number of users requesting file n be denoted as m_n , $n = 1, 2, \dots, N$. We call the vector obtained by sorting the values $\{m_1, m_2, \dots, m_N\}$ in a decreasing order as the demand type, denoted as \mathcal{T} .*

Proposition 3.1 implies that for optimal symmetric solutions, demands of the same type can always be satisfied with transmissions of the same rate, however, demands of different types may still require different rates. This observation is also important in setting up the linear program in the computer-aided approach outlined in the previous section. Because we are interested in the worst case transmission rate among all types of demands, in the symmetry-reduced LP, an additional variable needs to be introduced to constrain the transmission rates of all possible types.

For an (N, K) system, determining the number of demand types is closely related to the integer partition problem, which is the number of possible ways to write an integer K as the sum of positive integers. There is no explicit formula, but one can use a generator polynomial to compute it [21]. For several small (N, K) pairs, we list the demand types in Table 1.

It can be seen that when $N \leq K$, increasing N induces more demand types, but this stops when $N > K$; however, increasing K always induces more demand types. This suggests it might be easier to find solutions for a collection of cases with a fixed K and arbitrary N values, but more difficult for that of a fixed N and arbitrary K values. This intuition is partially confirmed with our results presented next.

4 Computational and Data-Driven Converse

Extending the computational approach developed in [15] and the problem symmetry, in this section we first establish complete characterizations for the optimal memory-transmission-rate tradeoff for $(N, K) = (3, 2)$ and $(N, K) = (4, 2)$. Based on these results and the known result for $(N, K) = (2, 2)$, we are able

to form a hypothesis regarding the optimal tradeoff for the case of $K = 2$. An analytical proof is then provided, which gives the complete characterization of the optimal tradeoff for the case of $(N, 2)$ caching systems. We then present a characterization of the optimal tradeoff for $(N, K) = (2, 3)$ and an outer bound for $(N, K) = (2, 4)$. These results also motivate a hypothesis on the optimal tradeoff for $N = 2$, which is subsequently proved analytically to yield a partial characterization. Note that since both M and R must be nonnegative, we do not explicitly state their non-negativity from here on.

4.1 The Optimal Tradeoff for $K = 2$

Our investigation starts with identifying the optimal tradeoff for $(N, K) = (3, 2)$ and $(N, K) = (4, 2)$ using the computation approach outline in Section 2, which are first summarized below as two propositions.

Proposition 4.1. *Any memory-transmission-rate tradeoff pair for the $(N, K) = (3, 2)$ caching problem must satisfy*

$$M + R \geq 2, \quad M + 3R \geq 3. \quad (18)$$

Conversely, there exist codes for any nonnegative (M, R) pair satisfying (18).

Proposition 4.2. *Any memory-transmission-rate tradeoff pair for the $(N, K) = (4, 2)$ caching problem must satisfy*

$$3M + 4R \geq 8, \quad M + 4R \geq 4. \quad (19)$$

Conversely, there exist codes for any nonnegative (M, R) pair satisfying (19).

The optimal tradeoff for $(N, K) = (2, 2)$ was found in [1], which we restated below.

Proposition 4.3 (Madduh Ali and Niesen [1]). *Any memory-transmission-rate tradeoff pair for the $(N, K) = (2, 2)$ caching problem must satisfy*

$$2M + R \geq 2, \quad 2M + 2R \geq 3, \quad M + 2R \geq 2. \quad (20)$$

Conversely, there exist codes for any nonnegative (M, R) pair satisfying (20).

The proofs for Proposition 4.1 and Proposition 4.2 can be found in the appendix, which are given in the tabulation format mentioned earlier. Strictly speaking, these two results are specialization of Theorem 4.5, and there is no need to provide the proofs separately, however we provide them to illustrate our computer-aided approach more explicitly.

The optimal tradeoff for the three cases are depicted in Fig. 2. A few immediate observations are as follows

- For $(N, K) = (3, 2)$ and $(N, K) = (4, 2)$, there is only one non-trivial corner point on the optimal tradeoff, but for $(N, K) = (2, 2)$ there are in fact two non-trivial corner points.
- The cut-set bound is tight at the high memory regime in all the cases.
- The single non-trivial corner point is achieved by the uncoded-placement-coded-transmission scheme proposed in [1]. For the $(N, K) = (2, 2)$ case, one of the corner point is achieved also by this scheme, but the other corner point requires a different coded-placement-uncoded-transmission coding strategy.
- For both $(N, K) = (3, 2)$ and $(N, K) = (4, 2)$, the non-trivial corner point has transmission rate $R = 1/2$, and the corresponding memory value as specified in Theorem 2.1 which is $N/2$.

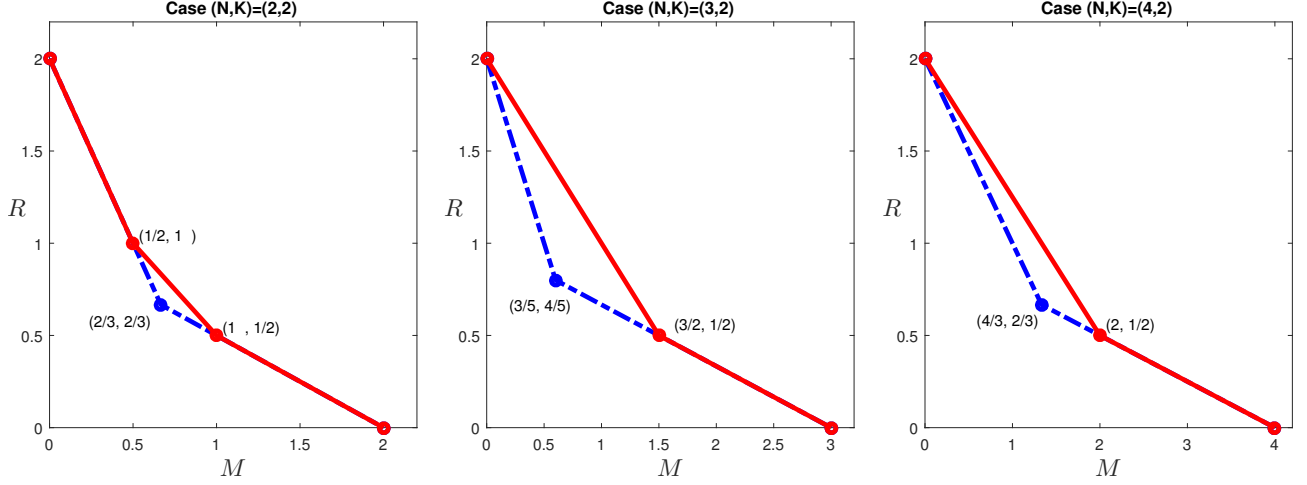


Figure 2: The optimal tradeoffs for $(N, K) = (2, 2)$, $(N, K) = (3, 2)$ and $(N, K) = (4, 2)$ caching systems. The red solid lines give the optimal tradeoffs, while the blue dashed-dot lines are the cut-set outer bounds, included here for reference.

Given the above observations, a natural hypothesis regarding the optimal tradeoff for $K = 2$ is as follows.

Hypothesis 4.4. *There is only one non-trivial corner point on the optimal tradeoff for $(N, K) = (N, 2)$ caching systems when $N \geq 3$, and it is $(M, R) = (N/2, 1/2)$, or equivalently the two facets of the optimal tradeoff should be*

$$3M + NR \geq 2N, \quad M + NR \geq N. \quad (21)$$

We are indeed able to analytically confirm this hypothesis, as stated formally in the following theorem.

Theorem 4.5. *For any integer $N \geq 3$, any memory-transmission-rate tradeoff pair for the $(N, K) = (N, 2)$ caching problem must satisfy*

$$3M + NR \geq 2N, \quad M + NR \geq N. \quad (22)$$

Conversely, for any integer $N \geq 3$, there exist codes for any nonnegative (M, R) pair satisfying (22). For $(N, K) = (2, 2)$, the memory-transmission-rate tradeoff must satisfy

$$2M + R \geq 2, \quad 2M + 2R \geq 3, \quad M + 2R \geq 2. \quad (23)$$

Conversely, for $(N, K) = (2, 2)$, there exist codes for any nonnegative (M, R) pair satisfying (23).

Since the solution for the special case $(N, K) = (2, 2)$ was established in [1], we only need to consider the cases for $N \geq 3$. Moreover, for the converse direction, only the bound $3M + NR \geq 2N$ needs to be proved, since the other one can be obtained using the cut-set bound in [1]. To prove the remaining inequality, the following auxiliary lemma is needed.

Lemma 4.6. *For any symmetric $(N, 2)$ caching code where $N \geq 3$, and any integer $n = \{1, 2, \dots, N-2\}$, we have*

$$\begin{aligned} (N-n)H(Z_1, W_{[1:n]}, X_{n,n+1}) \\ \geq (N-n-2)H(Z_1, W_{[1:n]}) + (N+n). \end{aligned} \quad (24)$$

Using Lemma 4.6, we can prove the converse part of Theorem 4.5 through an induction; the proofs of Theorem 4.5 and Lemma 4.6 can be found in the appendix, both of which heavily rely on the symmetry specified in the previous section. We note that although some clues can be found in the proof tables for the cases $(N, K) = (3, 2)$ and $(N, K) = (4, 2)$, such as the effective joint entropy terms in the converse proof, finding the proof of Theorem 4.5 still requires considerable human effort, and we were not able to complete this step directly through a computer program. One key observation simplifying the proof is that as the hypothesis states, the optimal corner point is achieved by the scheme given in [1]. In this specific case, the scheme reduces to splitting each file into half, and placing one half at the first user, and the other half at the second user; the corresponding delivery strategy is also extremely simple. We essentially combined this special structure and the clues from the proof tables to find the outer bounding steps.

Remark 4.7. *One ISIT reviewer pointed out to us that the result in [5] can be used to establish the bound $3M + NR \geq 2N$, however only for the cases when N is an integer multiple of 3. For $N = 4$, the bounds developed in [5–7] give $M + 2R \geq 3$, instead of $3M + 4R \geq 8$, and thus they are loose in this case.*

4.2 A Partial Characterization for $N = 2$

We first summarize the characterizations of the optimal tradeoff for $(N, K) = (2, 3)$, and the computed outer bound for $(N, K) = (2, 4)$, in two propositions.

Proposition 4.8. *The memory-transmission-rate tradeoff for the $(N, K) = (2, 3)$ caching problem must satisfy:*

$$2M + R \geq 2, \quad 3M + 3R \geq 5, \quad M + 2R \geq 2. \quad (25)$$

Conversely, there exist codes for any nonnegative (M, R) pair satisfying (25).

Proposition 4.9. *The memory-transmission-rate tradeoff for the $(N, K) = (2, 4)$ caching problem must satisfy:*

$$2M + R \geq 2, \quad 14M + 11R \geq 20, \quad 9M + 8R \geq 14, \quad 3M + 3R \geq 5, \quad 5M + 6R \geq 9, \quad M + 2R \geq 2. \quad (26)$$

For Proposition 4.8, the only new bound $3M + 3R \geq 5$ is a special case of the more general result of Theorem 4.12 and we thus do not provide this proof separately. For Proposition 4.9, only the second and the third inequalities need to be proved, since the fourth coincides with a bound in the $(2, 3)$ case, the fifth is a special case of Theorem 4.12, and the others can be produced from the cut-set bounds. The proofs for these two inequalities are omitted here, but the proof data file is included as a supplement to the paper, and is also made online at [22]. The optimal tradeoff for $(N, K) = (2, 2)$, $(2, 3)$ and the outer bound for $(2, 4)$ are depicted in Fig. 3. A few immediate observations and comments are as follows:

- There are two non-trivial corner points on the outer bounds for $(N, K) = (2, 2)$ and $(N, K) = (2, 3)$, and there are five non-trivial corner points for $(N, K) = (2, 4)$.
- The outer bounds coincide with known inner bounds for $(N, K) = (2, 2)$ and $(N, K) = (2, 3)$, but not $(N, K) = (2, 4)$. The corner points at the high memory value $R = 1/K$ (and the corner point $(1, 2/3)$ for $(N, K) = (2, 4)$) are achieved by the scheme given in [1], while the corner points at the low memory value $M = 1/K$ are achieved by the scheme given in [8]. For $(N, K) = (2, 4)$, two corner points at the intermediate memory regime cannot be achieved by either the scheme in [1] or that in [8]. This suggests that at the intermediate memory regime, known schemes are not sufficient, and new codes are needed. Indeed in the next section we discuss a new code to achieve one of the corner points that were previously not known to be achievable.

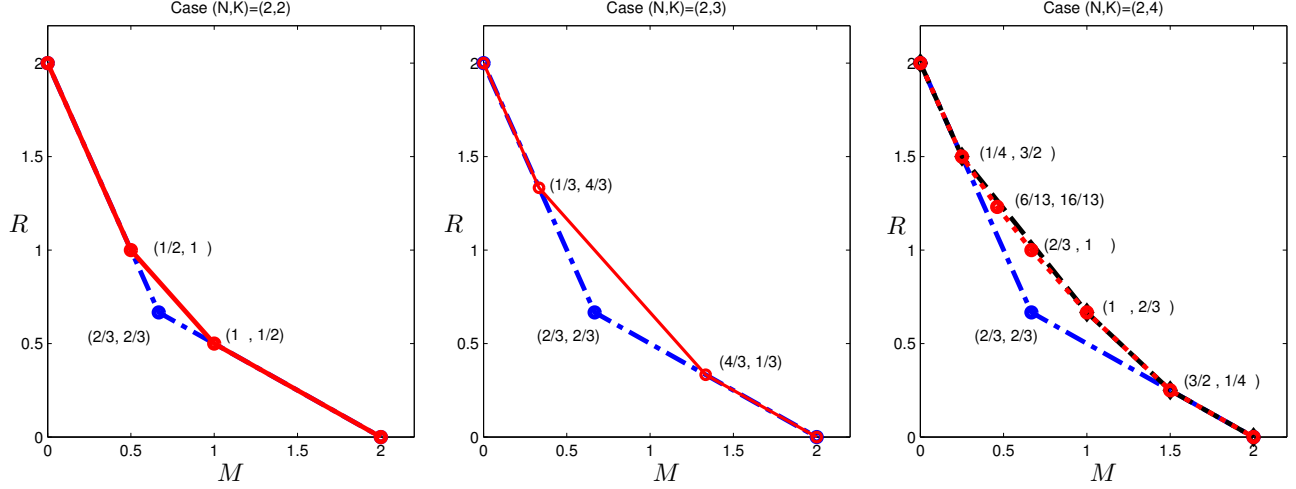


Figure 3: The optimal tradeoffs for $(N, K) = (2, 2)$, $(N, K) = (2, 3)$ and computed outer bound $(N, K) = (2, 4)$ caching systems. The red solid lines give the optimal tradeoffs for the first two case, and the red dotted line gives the computed outer bound $(N, K) = (2, 4)$; The blue dashed-dot lines are the cut-set outer bounds, and the black dashed line is the inner bound using the scheme in [1] and [8].

- The cut-set outer bounds [1] are tight at the highest and lowest memory segments; a new bound for the second highest memory segment produced by the computer based method is also tight.

Remark 4.10. The bounds developed in [5–7] give $2(M + R) \geq 3$ for $(N, K) = (2, 3)$ and $(N, K) = (2, 4)$, instead of $3M + 3R \geq 5$, and thus they are loose in this case.

From the above observations, we can hypothesize that for $N = 2$, the number of corner points will continue to increase as K increases above 4, and at the high memory regime, the scheme [1] is optimal. More precisely, the following hypothesis appear to be a natural first step.

Hypothesis 4.11. The first two non-trivial (M, R) corner points of the optimal tradeoff for $(N, K) = (2, K)$ at the high memory regime when $K \geq 4$ are

$$\left(\frac{2(K-1)}{K}, \frac{1}{K} \right) \quad \text{and} \quad \left(\frac{2(K-2)}{K}, \frac{2}{K-1} \right). \quad (27)$$

Conversely, when $K \geq 4$ and $N = 2$, any (M, R) pair must satisfy

$$K(K+1)M + 2(K-1)KR \geq 2(K-1)(K+2). \quad (28)$$

This following theorem confirms that the hypothesis is indeed true.

Theorem 4.12. When $K \geq 3$ and $N = 2$, any (M, R) pair must satisfy

$$K(K+1)M + 2(K-1)KR \geq 2(K-1)(K+2). \quad (29)$$

As a consequence, the uncoded-placement-coded-transmission scheme in [1] (with space-sharing) is optimal when $M \geq \frac{2(K-2)}{K}$, for the cases with $K \geq 4$ and $N = 2$.

The first line segment at the high memory regime is $M + 2R \geq 2$, which is given by the cut-set bound; its intersection with (29) is indeed

$$\left(\frac{2(K-1)}{K}, \frac{1}{K} \right). \quad (30)$$

The proof of this theorem now boils down to the proof of the bound (29). This requires a sophisticated induction, the digest of which is summarized in the following lemma. The symmetry of the problem is again heavily utilized throughout of the proof of this lemma. For notational simplicity, we use $X_{\rightarrow j}$ to denote $X_{1,1,\dots,1,2,1,\dots,1}$, *i.e.*, when the j -t user requests the second file, and all the other users request the first file; we also write a collection of such variables $(X_{\rightarrow j}, X_{\rightarrow j+1}, \dots, X_{\rightarrow k})$ as $X_{\rightarrow [j:k]}$.

Lemma 4.13. *For $N = 2$ and $K \geq 3$, the following inequality holds for $k \in \{2, 3, \dots, K - 1\}$*

$$\begin{aligned} & (K - k + 1)(K - k + 2)H(Z_1, W_1, X_{\rightarrow [2:k]}) \\ & \geq [(K - k)(K - k + 1) - 2]H(Z_1, W_1, X_{\rightarrow [2:k-1]}) + 2H(W_1, X_{\rightarrow [2:k-1]}) + 2(K - k + 1)H(W_1, W_2), \end{aligned} \quad (31)$$

where we have taken the convention $H(Z_1, W_1, X_{\rightarrow [2:1]}) = H(Z_1, W_1)$

The proof of Lemma 4.13 can be found in the appendix. Theorem 4.12 can now be proved straightforwardly.

Proof of Theorem 4.12. We first write the following simple inequalities

$$H(Z_1) + H(X_{\rightarrow 2}) \geq H(Z_1, X_{\rightarrow 2}) = H(Z_1, W_1, X_{\rightarrow 2}). \quad (32)$$

Now applying Lemma 4.13 with $k = 2$ gives

$$\begin{aligned} & (K - 1)K[H(Z_1) + H(X_{\rightarrow 2})] \\ & \geq [K^2 - 3K]H(Z_1, W_1) + 2H(W_1) + 2(K - 1)H(W_1, W_2). \end{aligned} \quad (33)$$

Observe that

$$H(Z_1, W_1) = H(W_1|Z_1) + H(Z_1) \geq \frac{1}{2}H(W_1, W_2|Z_1) + H(Z_1) = \frac{1}{2}H(W_1, W_2) + \frac{1}{2}H(Z_1), \quad (34)$$

where in the first inequality the file index symmetry $H(W_1|Z_1) = H(W_2|Z_1)$ has been used. We can now continue to write

$$\begin{aligned} & (K - 1)K[H(Z_1) + H(X_{\rightarrow 2})] \\ & \geq \frac{K^2 - 3K}{2}[H(W_1, W_2) + H(Z_1)] + 2H(W_1) + 2(K - 1)H(W_1, W_2), \end{aligned} \quad (35)$$

which has some a common term $H(Z_1)$ on both sides with different coefficients. Removing the common term and multiplying both sides by two lead to

$$\begin{aligned} & K(K + 1)H(Z_1) + 2(K - 1)KH(X_{\rightarrow 2}) \\ & \geq [(K - 2)(K - 1) - 2 + 4(K - 1)]H(W_1, W_2) + 4H(W_1) \\ & = 2K^2 + 2K - 4, \end{aligned} \quad (36)$$

where the equality relies on the assumption that W_1 and W_2 are independent files of unit size. Taking into consideration the memory and transmission rate constraints (4) and (5) now completes the proof. \square

Lemma 4.13 provides a way to reduce the number of X variables in $H(Z_1, X_{\rightarrow [2:k]})$, and thus is the core of the proof. Readers may observe that Lemma 4.13 is only used for the case $k = 2$ in the proof of the theorem, however, the lemma is stated in a more general manner. We chose to present the lemma this way because the proof for the case $k = 2$ in fact requires the inequalities for other cases, and we suspect this lemma (or a lemma in a similar form) can also be useful in proving outer bounds for other segments of the optimal tradeoff.

Table 2: Caching content for $(N, K) = (2, 4)$

User 1	$A_1 + B_1$	$A_2 + B_2$	$A_3 + B_3$	$A_1 + A_2 + A_3 + 2(B_1 + B_2 + B_3)$
User 2	$A_1 + B_1$	$A_4 + B_4$	$A_5 + B_5$	$A_1 + A_4 + A_5 + 2(B_1 + B_4 + B_5)$
User 3	$A_2 + B_2$	$A_4 + B_4$	$A_6 + B_6$	$A_2 + A_4 + A_6 + 2(B_2 + B_4 + B_6)$
User 4	$A_3 + B_3$	$A_5 + B_5$	$A_6 + B_6$	$A_3 + A_5 + A_6 + 2(B_3 + B_5 + B_6)$

Even with the hypothesis regarding the scheme in [1] being optimal, deriving the outer bound (particularly the coefficients in the lemma above) directly using this insight is far from being straightforward. Some of the guidance in finding our derivation was in fact obtained through a strategic computational exploration of the outer bounds. This information is helpful because the computer-generated proofs are not unique, and some of these solutions can appear quite arbitrary, however, to deduce general rules in the proof requires a more structured proof instead. In Section 6, we present in more details this new exploration method, and shall revisit this issue regarding the “helpful” information in this particular case as an illustrative example.

We remark here that although we focused on the memory range $M \geq \frac{2(K-2)}{K}$ for $N = 2$, the computed data collected so far suggests that the uncoded-placement-code-transmission scheme given in [1] may be optimal for more general parameters, particularly at the high memory regime; in a subsequent work, we explore this direction further, and will present characterizations for more general classes of problem cases in a forthcoming paper.

5 Reverse-Engineering Code Constructions

In the previous section, outer bounds of the optimal tradeoff were presented for the case $(N, K) = (2, 4)$, which is given in Fig. 3. Observe that the corner points

$$\left(\frac{2}{3}, 1\right) \quad \text{and} \quad \left(\frac{6}{13}, \frac{16}{13}\right), \quad (37)$$

cannot be achieved by existing codes in the literature. The former point can indeed be achieved with a new code construction. This construction was first presented in [12], where it was generalized more systematically to yield a new class of codes for any $N \leq K$, whose proof and analysis are more involved. In this paper, we focus on how a specific code for this corner point can be found through a reverse engineering approach, which should help dispel the mystery on this seemingly arbitrary code construction. This section is organized as follows: first the new code is presented, then the method to extract the related information in the LP is discussed, the data which leads to this code construction is then given and analyzed, and finally we present an example where this reverse engineering approach can be used to show linear codes cannot be constructed for a given memory-transmission-rate tradeoff pair.

5.1 The Code to Achieve $\left(\frac{2}{3}, 1\right)$ for $(N, K) = (2, 4)$

The two files are denoted as A and B , each of which is partitioned into 6 segments of equal size, denoted as A_i and B_i , respectively, $i = 1, 2, \dots, 6$. Since we count the memory and transmission in multiple of the file size, the corner point $\left(\frac{2}{3}, 1\right)$ means needs each user to store 4 symbols, and the transmission will use 6 symbols. The contents in the cache of each user are given in Table 2. By the symmetry of the cached contents, we only need to consider the demand (A, A, A, B) , *i.e.*, the first three users requesting A and user 4 requesting B , and the demand (A, A, B, B) , *i.e.*, the first two users requesting A and the other two requesting B . Assume the file segments are in \mathbb{F}_5 for concreteness.

- For the demands (A, A, A, B) , the transmission is as follows,

Step 1: B_1, B_2, B_4 ;

Step 2: $A_3 + 2A_5 + 3A_6, A_3 + 3A_5 + 4A_6$;

Step 3: $A_1 + A_2 + A_4$.

After step 1, user 1 can recover (A_1, A_2) ; furthermore, he has $(A_3 + B_3, A_3 + 2B_3)$ by eliminating known symbols (A_1, A_2, B_1, B_2) , from which A_3 can be recovered. After step 2, he can obtain $(2A_5 + 3A_6, 3A_5 + 4A_6)$ to recover (A_5, A_6) . Using the transmission in step 3, he can obtain A_4 since he has (A_1, A_2) . User 2 and user 3 can use a similar strategy to reconstruct all file segments in A . User 4 only needs B_3, B_5, B_6 after step 1, which he already has in his cache, however they are contaminated by file segments from A . Nevertheless, he knows $A_3 + A_5 + A_6$ by recognizing

$$(A_3 + A_5 + A_6) = 2 \sum_{i=3,5,6} (A_i + B_i) - [A_3 + A_5 + A_6 + 2(B_3 + B_5 + B_6)]. \quad (38)$$

Together with the transmission in step 2, user 4 has three linearly independent combinations of (A_3, A_5, A_6) . After recovering them, he can remove these interferences from the cached content for (B_3, B_5, B_6) .

- For the demand (A, A, B, B) , we can send

Step 1: B_1, A_6 ;

Step 2: $A_2 + 2A_4, A_3 + 2A_5, B_2 + 2B_3, B_4 + 2B_5$.

User 1 has A_1, B_1, A_6 after step 1, and he can also form

$$B_2 + B_3 = [A_2 + A_3 + 2(B_2 + B_3)] - (A_2 + B_2) - (A_3 + B_3),$$

and together with $B_2 + 2B_3$ in the transmission of step 2, he can recover (B_2, B_3) , and thus A_2, A_3 . He still needs (A_4, A_5) , which can be recovered straightforwardly from the transmission $(A_2 + 2A_4, A_3 + 2A_5)$ since he already has (A_2, A_3) . Other users can use a similar strategy to decode their requested files.

5.2 Extracting Information for Reverse Engineering

It is clear at this point that even for this simple case of $(N, K) = (2, 4)$, the code to achieve this optimal corner point is not straightforward. Next we discuss a general approach to deduce the code structure from the LP solution, which leads to the discovery of the code in our work. The approach is based on the following assumptions: the outer bound is achievable (*i.e.*, tight), moreover, there is a (vector) linear code that can achieve this performance.

Either of the two assumptions above may not be hold in general, and in such a case our attempt will not be successful. Nevertheless, though linear codes are known to be not sufficient for all network coding problem [23], existing results in the literature suggest that vector linear codes are surprisingly versatile and powerful. Similarly, though it is known that Shannon-type inequalities, which are the basis for the outer bounds computation, are not sufficient to characterize rate region for all coding problems [24, 25], they are surprisingly powerful, particularly in coding problems with strong symmetry structures [26, 27].

There are essentially two types of information that we can extract from the primal LP and dual LP:

- From the effective information inequalities: since we can produce a readable proof using the dual LP, if a code can achieve this corner point, then the information inequalities in the proof must hold with equality for the joint entropy values induced by this code, which reveals a set of conditional independence relations among random variables induced by this code;
- From the extremal joint entropy values at the corner points: although we are only interested in the tradeoff between the memory and transmission rate, the LP solution can provide the whole set of joint entropy values at an extreme point. These values can reveal a set of dependence relations among the random variables induced by any code that can achieve this point.

Though the first type of information is important, its translation to code constructions appears difficult. On the other hand, the second type of information appears to be more suitable for the purpose of code design, which we adopt next.

One issue that complicates our task is that the entropy values such extracted are not always unique, and sometimes have considerable slacks. For example, for different LP solutions at the same operating point of $(M, R) = (\frac{2}{3}, 1)$, the joint entropy $H(Z_1, Z_2)$ can vary between 1 and $4/3$. We can identify such a slack in any joint entropy in the corner point solutions by considering a regularized primal LP: for a fixed rate value R at the corner point in question as an upper bound, the objective function can be set as

$$\text{minimize: } H(Z_1) + \gamma H(Z_1, Z_2) \quad (39)$$

instead of

$$\text{minimize: } H(Z_1), \quad (40)$$

subject to the same original symmetric LP constraints at the target M . By choosing a small positive γ value, *e.g.*, $\gamma = 0.0001$, we can find the minimum value for $H(Z_1, Z_2)$ at the same (M, R) point; similarly, by choosing a small negative γ value, we can find the maximum value for $H(Z_1, Z_2)$ at the same (M, R) point. Such slacks in the solution add uncertainty to the codes we seek to find, and may indeed imply the existence of multiple code constructions. Nevertheless, for the purpose of reverse engineering the codes, we shall focus on the joint entropies that do not have any slacks using this procedure, *i.e.*, the “stable” joint entropies in the solution. In a sense, such stable solutions reflects the critical coding constraints and the structure that must be maintained to construct a valid code, while the slacks provide further flexibility in the code design space.

5.3 Reverse-Engineering the Code for $(N, K) = (2, 4)$

With the method outlined above, we identify the following stable joint entropy values in the $(N, K) = (2, 4)$ case for the operating point $(\frac{2}{3}, 1)$ listed in Table 3. We have normalized the values by multiplying everything by 6; the corresponding conditional entropies are given in the last column. For simplicity, let us assume that each file has 6 units of information, written as $W_1 = (A_1, A_2, \dots, A_6) \triangleq A$ and $W_2 = (B_1, B_2, \dots, B_6) \triangleq B$, respectively. This is a rich set of data, but a few immediate observations are given follows. It should be noted that each step in this reverse-engineering argument may be only sufficient but not necessary. However, since the eventual goal is to find a code to achieve that operating point, we only need to assure the validity of the end result.

- The quantities can be categorized into three groups: the first group is without any transmission, the second group is the quantities involving the transmission to fulfill the demand type $(3, 1)$, and the last group is for demand type $(2, 2)$.
- The conditional versions of the first three quantities $H(Z_1|W_1)$, $H(Z_1, Z_2|W_1)$ and $H(Z_1, Z_2, Z_3|W_1)$ provide the most important clue. The values indicate that for each of the two files, each user should have 3 units in its cache, and the combination of any two users should have 5 units in their cache,

Table 3: Stable joint entropy values at the corner point $(\frac{2}{3}, 1)$ for $(N, K) = (2, 4)$.

Joint entropy	Value*6	$H(\cdot W_1)$
$H(W_1, Z_1)$	9	3
$H(W_1, Z_1, Z_2)$	11	5
$H(W_1, Z_1, Z_2, Z_3)$	12	6
$H(W_1, X_{1,2,2,2})$	9	3
$H(W_1, Z_1, X_{1,2,2,2})$	10	4
$H(W_1, X_{1,1,1,2})$	9	3
$H(W_1, Z_1, X_{1,1,1,2})$	10	4
$H(W_1, Z_1, Z_2, X_{1,1,1,2})$	11	5
$H(W_1, Z_1, Z_2, Z_3, X_{1,1,1,2})$	12	6
$H(W_1, X_{1,1,1,2}, X_{1,1,2,1})$	11	5
$H(W_1, Z_1, X_{1,1,1,2}, X_{1,1,2,1})$	11	5
$H(W_1, Z_1, Z_2, X_{1,1,1,2}, X_{1,1,2,1})$	11	5
$H(W_1, X_{1,1,2,2})$	9	3
$H(W_1, Z_1, X_{1,1,2,2})$	10	4
$H(W_1, Z_1, Z_2, X_{1,1,2,2})$	11	5

and the combination of any three users should have all 6 units in their cache. This strongly suggests placing each piece A_i (and B_i) at two users. Since each Z_i has 4 units, but it needs to hold 3 units from each of the two files, coded placement (cross files) is thus needed. At this point, we can place the corresponding symbols in the caching, but keep the precise linear combination coefficients as undetermined.

- The next critical observation is that $H(X_{1,2,2,2}|W_1) = H(X_{1,1,1,2}|W_1) = H(X_{1,1,2,2}|W_1) = 3$. This implies that the transmission has 3 units of information on each file alone. However, since the operating point dictates that $H(X_{1,2,2,2}) = H(X_{1,1,1,2}) = H(X_{1,1,2,2}) = 6$, it further implies that in each transmission, 3 units are for the linear combinations of W_2 , and 3 units are for those of W_1 ; in other words, the linear combinations do not mix information from different files.
- Since each transmission only has 3 units of information from each file, and each user has only 3 units of information from each file, they must be linearly independent of each other.

The observation and deductions are only from the perspective of the joint entropies given in Table 3, without much consideration of the particular coding requirement. For example, in the last item discussed above, it is clear that when transmitting the 3 units of information regarding a file (say file W_2), they should be simultaneously useful to other users requesting this file, and to the users not requesting this file. This intuition then strongly suggests each transmitted linear combination of W_2 should be a subspace of the W_2 parts at some users not requesting it. Using these intuitions as guidance, finding the code becomes straightforward after a few trial-and-errors. In [12] we were able to further generalize this special code to a class of codes for any case when $N \leq K$; readers are referred to [12] for more details on these codes.

5.4 Disproving Linear-Coding Achievability

The reverse engineering approach may not always be successful, either because the structure revealed by the data is very difficult to construct explicitly, or because linear codes are not sufficient to achieve this operating point. For instance, we were not able to explicitly construct a code for the other corner

Table 4: Stable joint entropy values at the corner point $(\frac{2}{3}, \frac{4}{3})$ for $(N, K) = (3, 3)$.

Joint entropy	$H(\cdot W_1)$
$H(Z_1 W_1)$	$2m$
$H(Z_1 W_1, W_2)$	m
$H(Z_1, Z_2 W_1, W_2)$	$2m$
$H(Z_1, Z_2, Z_3 W_1, W_2)$	$3m$
$H(X_{1,2,3})$	$4m$
$H(X_{1,2,3} W_1)$	$3m$
$H(X_{1,2,3} W_1, W_2)$	$2m$

point $(6/13, 16/13)$ for the $(N, K) = (2, 4)$ case, and it is not clear whether it is due to the first or the second reason. In some other cases, the determination can be done explicitly. In the sequel we present an example for $(N, K) = (3, 3)$, which belongs to the latter case.

In the next section, we provide an outer bound for $(N, K) = (3, 3)$. Among the corner points of the outer bound, the pair $(M, R) = (\frac{2}{3}, \frac{4}{3})$ is the only one that cannot be achieved by existing schemes. Since the outer bound appears quite strong, we suspect this pair is also achievable and thus attempt to construct a corresponding linear code. Unfortunately, as we shall show next, there does not exist such a (vector) linear code. Before delving into the data provided by the LP, readers are encouraged to consider proving directly that this tradeoff point cannot be achieved by linear codes, which does not appear to be straightforward to the author.

We shall assume each file has $3m$ symbols in certain finite field, where m is a positive integer. The LP produces the joint entropy values (in terms of the number of finite field symbols, not in multiples of file size as in the other sections of the paper) in Table. 4 at this corner point, where only the conditional joint entropies relevant to our discussion next are listed.

The first critical observation is that $H(Z_1|W_1, W_2) = m$, and the user-index-symmetry implies that $H(Z_2|W_1, W_2) = H(Z_3|W_1, W_2) = m$. Moreover $H(Z_1, Z_2, Z_3|W_1, W_2) = 3m$, from which we can conclude that excluding file W_1 and W_2 , each user stores m linearly independent combinations of the symbols of file W_3 . Similar conclusions hold for files W_1 and W_2 . This implies that through a change of basis for each file, we can assume without loss of generality that user- k stores $2m$ linear combinations in the following form

$$V_k \cdot \begin{bmatrix} W_{1,[(k-1)m+1:km]} \\ W_{2,[(k-1)m+1:km]} \\ W_{3,[(k-1)m+1:km]} \end{bmatrix} \quad (41)$$

where $W_{n,j}$ is the j -th symbol of the n -th file, and V_k is a matrix of dimension $2m \times 3m$; V_k can be partitioned into submatrices of dimension $m \times m$, which are denoted as $V_{k;i,j}$, $i = 1, 2$ and $j = 1, 2, 3$. Note that symbols at different users are orthogonal to each other without loss of generality.

Without loss of generality, assume the transmitted content $X_{1,2,3}$ is

$$G \cdot \begin{bmatrix} W_{1,[1:3m]} \\ W_{2,[1:3m]} \\ W_{3,[1:3m]} \end{bmatrix} \quad (42)$$

where G is a matrix of dimension $4m \times 9m$; we can partition it into blocks of $m \times m$, and each block is referred to as $G_{i,j}$, $i = 1, 2, \dots, 4$ and $j = 1, 2, \dots, 9$. Let us first consider user 1, which has the following

symbols

$$\begin{bmatrix} V_{k;1,1} & 0 & 0 & V_{k;1,2} & 0 & 0 & V_{k;1,3} & 0 & 0 \\ V_{k;2,1} & 0 & 0 & V_{k;2,2} & 0 & 0 & V_{k;2,3} & 0 & 0 \\ \hline G_{1,1} & G_{1,2} & & \dots & & & & G_{1,9} \\ \vdots & \vdots & & \vdots & & & & \vdots \\ G_{4,1} & G_{4,2} & & \dots & & & & G_{4,9} \end{bmatrix} \cdot \begin{bmatrix} W_{1,[1:3m]} \\ W_{2,[1:3m]} \\ W_{3,[1:3m]} \end{bmatrix} \quad (43)$$

The coding requirement states that $X_{1,2,3}$ and Z_1 together can be used to recover file W_1 , and Table 4 specifies $H(Z_1|W_1) = 2m$, thus after removing W_1 from $X_{1,2,3}$, we have full rank linear combinations of $W_{2,[1:m]}$ and $W_{3,[1:m]}$ which can be used to recover these two vectors. It follows that through elemental row operations and column permutation, the matrix in (43) can be converted into the following form

$$\begin{bmatrix} U_{1,1} & U_{1,2} & U_{1,3} & 0 & & & & & 0 \\ U_{2,1} & U_{2,2} & U_{2,3} & 0 & & & & & 0 \\ U_{3,1} & U_{3,2} & U_{3,3} & 0 & & & & & 0 \\ 0 & 0 & 0 & U_{4,4} & U_{5,7} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & U_{4,4} & U_{5,7} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & U_{6,5} & U_{6,6} & U_{6,8} & U_{6,9} \end{bmatrix} \cdot \begin{bmatrix} W_{1,[1:3m]} \\ W_{2,[1:m]} \\ W_{3,[1:m]} \\ W_{2,[m+1:3m]} \\ W_{3,[m+1:3m]} \end{bmatrix}, \quad (44)$$

where diagonal block matrices are of full rank $3m$ and $2m$, respectively. The matrix $[U_{6,5}, U_{6,6}, U_{6,8}, U_{6,9}]$ has maximum rank m , which implies that the matrix

$$\begin{bmatrix} G_{1,5} & G_{1,6} & G_{1,8} & G_{1,9} \\ \vdots & \vdots & \vdots & \vdots \\ G_{4,5} & G_{4,6} & G_{4,8} & G_{4,9} \end{bmatrix}, \quad (45)$$

i.e., the submatrix of G by taking columns (5, 6, 8, 9), has only maximum rank m . However, due to the symmetry, we can also conclude that the submatrix of G taking only columns (1, 3, 7, 9) and that taking only columns (1, 2, 4, 5) both have only maximum rank m . This further implies that the matrix G has rank no larger than $3m$, but this does not satisfy the condition that $H(X_{1,2,3}) = 4m$ in Table 4. We can now conclude that this memory-transmission-rate pair is not achievable with any linear codes².

6 Computational Exploration and Bounds for Larger Cases

In this section we explore the fundamental limits of the caching systems in more details using a computational approach. Due to the (doubly) exponential growth of the LP variables and constraints, directly applying the method outlined in Section 2 becomes infeasible for larger problem cases. This is the initial motivation for us to investigate single-demand-type systems where only a single demand type is allowed. Any outer bound on the tradeoff of such a system is an outer bound for the original one, and the intersection of these outer bounds is thus also an outer bound. This investigation further reveals several hidden phenomena. For example, outer bounds for different single-demand-type systems are stronger in different regimes, and moreover, the LP bound for the original system is not simply the intersection of all outer bounds for single-demand-type systems, but in certain regimes they do match.

Given the observations above, we take the investigation one step further by choosing only a small subset of demands instead of the complete set in a single demand type. This allows us to obtain results

²Strictly speaking, our argument above holds under the assumption that the joint entropy values produced by LP are precise rational values, and the machine precision issue has thus been ignored. However, if the solution is accurate only up to machine precision, one can introduce a small slack value δ into the quantities, *e.g.*, replacing $3m$ with $(3 \pm \delta)m$, and using a similar argument show that the same conclusion holds. This extended argument however becomes notationally rather lengthy, and we thus omitted it here for simplicity.

for cases which initially appear impossible to compute. For example, even for $(N, K) = (2, 5)$, there are a total of $2 + 5 + 2^5 = 39$ random variables, and the number of constraints in LP after symmetry reduction is more than 10^{11} , which is significantly beyond current LP solver capability³. However, by strategically considering only a small subset of the demand patterns, we are indeed able to find meaningful outer bounds, and moreover, use the clues obtained in such computational exploration to complete the proof of Theorem 4.12. We shall discuss the method we develop, and also present several example results for larger problem cases. Clearly, this approach enables us to obtain outer bounds for a wider range of problem parameters, and furthermore opens a door to a possible large scale data-driven investigation, which is part of our ongoing work.

6.1 Single-Demand-Type Systems

As mentioned above, in a single-demand-type caching systems, the demand must belong to a particular demand type. We first present results on two cases $(N, K) = (2, 4)$ and $(N, K) = (3, 3)$, and then discuss our observations using these results.

Proposition 6.1. *Any memory-transmission-rate tradeoff pair for the $(N, K) = (2, 4)$ caching problem must satisfy the following conditions for **single demand type** $(4, 0)$:*

$$M + 2R \geq 2, \quad (46)$$

*and conversely any non-negative (M, R) pair satisfying (46) is achievable for single demand type $(4, 0)$; it must satisfy for **single demand type** $(3, 1)$:*

$$2M + R \geq 2, \quad 8M + 6R \geq 11, \quad 3M + 3R \geq 5, \quad 5M + 6R \geq 9, \quad M + 2R \geq 2, \quad (47)$$

*and conversely any non-negative (M, R) pair satisfying (47) is achievable for single demand type $(3, 1)$; it must satisfy for **single demand type** $(2, 2)$*

$$2M + R \geq 2, \quad 3M + 3R \geq 5, \quad M + 2R \geq 2, \quad (48)$$

and conversely any non-negative (M, R) pair satisfying (48) is achievable for single demand type $(2, 2)$.

The optimal (M, R) tradeoffs are illustrated in Fig. 4 with the known inner bound, *i.e.*, those in [1, 8] and the one given in the last section, and the computed out bound of the original problem given in Section 4. Here the demand type $(3, 1)$ in fact provides the tightest outer bound which matches the known inner bound for $M \in [0, 1/4] \cup [2/3, 2]$. The converse proofs of (47) and (48) are obtained computationally, whose proof data files are included as supplement to the paper (in fact only the middle three inequalities in (47) and the second inequality in (48) need to be proved), and are also made available online at [22]. Although the original caching problem requires codes that can handle all types of demands, the optimal codes for single demand type systems turn out to be quite interesting by its own right, and thus we provide the forward proof of Theorem 6.1 in the appendix.

We in fact could not complete the LP computation for $(N, K) = (3, 3)$ with all the possible demands. There are a total of 33 random variables, and even after reduction the problem appears too large for the state-of-the-art LP solvers. However, the single-demand-type systems are simpler and outer bounds can indeed be obtained, which are summarized below; the proof data files are included as supplement to the paper, and are also be made available online at [22].

Proposition 6.2. *Any memory-transmission-rate tradeoff pair for the $(N, K) = (3, 3)$ caching problem must satisfy the following conditions for **single demand type** $(3, 0, 0)$:*

$$M + 3R \geq 3, \quad (49)$$

³The problem can be further reduced using problem specific implication structures as outlined in Section 2, but our experience suggests that even with such additional reduction the problem is still too large for a start-of-the-art LP solver.

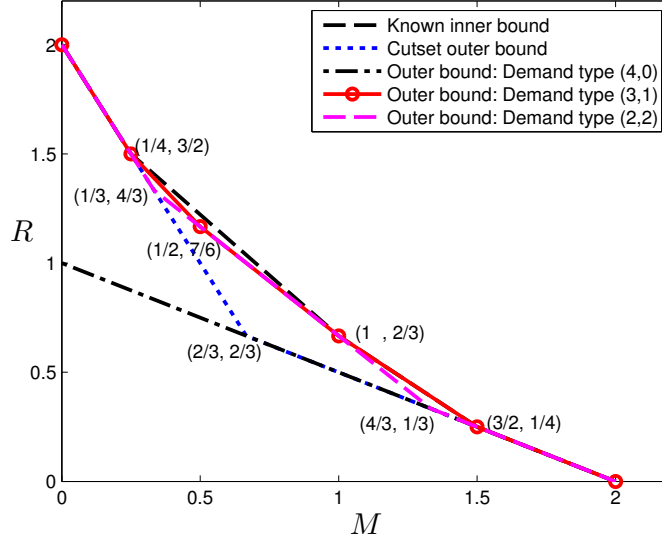


Figure 4: Tradeoff outer bounds for $(N, K) = (2, 4)$ caching systems.

and conversely any non-negative (M, R) pair satisfying (49) is achievable for single demand type $(3, 0, 0)$; it must satisfy for **single demand type** $(2, 1, 0)$:

$$M + R \geq 2, \quad 2M + 3R \geq 5, \quad M + 3R \geq 3, \quad (50)$$

and conversely any non-negative (M, R) pair satisfying (50) is achievable for single demand type $(2, 1, 0)$; it must satisfy for **single demand type** $(1, 1, 1)$:

$$3M + R \geq 3, \quad 6M + 3R \geq 8, \quad M + R \geq 2, \quad 12M + 18R \geq 29, \quad 3M + 6R \geq 8, \quad M + 3R \geq 3. \quad (51)$$

These outer bounds are illustrated in Fig. 5, together with the best known inner bound by combining [1] and [8], and the cut-set outer bound for reference. The bound is in fact tight for $M \in [0, 1/3] \cup [1, 3]$. Readers may notice that Proposition 6.2 provides complete characterizations for the first two demand types, but not the last demand type. As we have shown in Section 5, the point $(\frac{2}{3}, \frac{4}{3})$ in fact cannot be achieved using linear codes.

Remark 6.3. The bound developed in [6] gives $6M + 3R \geq 8$ and $2M + 4R \geq 5$, and that in [7] gives $(M + R) \geq 2$ in addition to the cut-set bound.

We can make the following observations immediately:

- The single-demand-type systems where few files are requested usually produce tighter bounds at high memory regime, while those request more files usually produce tighter bounds at low memory regime. For example, in the high memory regime, the first segment of the bounds can be obtained by considering only demands that request a single file which coincidentally is also the cut-set bound in this regime; for $(N, K) = (3, 3)$, the bound obtained from the demand type $(2, 1, 0)$ is stronger than that from $(1, 1, 1)$ in the range $M \in [1, 2]$.
- Simply intersecting the single-demand-type outer bounds does not produce the same bound as that obtained from a system with the complete set of demands. This can be seen from the case $(N, K) = (2, 4)$ in the range $M \in [1/4, 2/3]$.

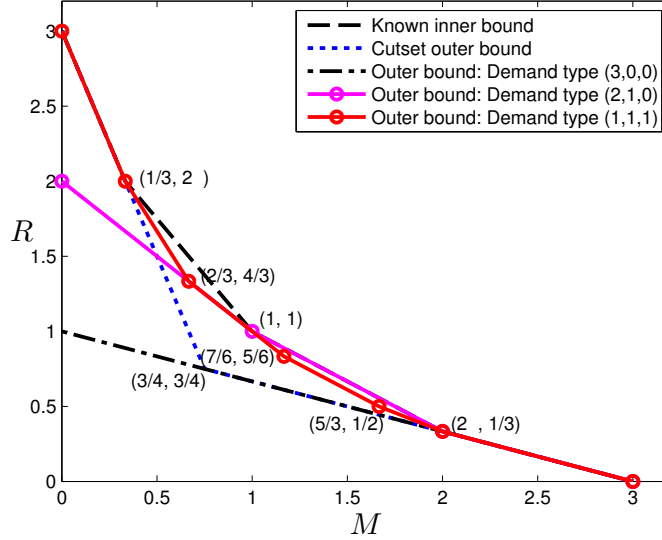


Figure 5: Tradeoff outer bounds for $(N, K) = (3, 3)$ caching

- The outer bounds produced by single-demand-type systems in many cases match the bound when more comprehensive demands are considered. This is particularly evident in the case $(N, K) = (2, 4)$ in the range $M \in [0, 1/4] \cup [2/3, 2]$.

These observations provide further insights on the difficulty of the problem. For instance, for $(N, K) = (2, 4)$, the demand type $(3, 1)$ is the most demanding case, and code design for this demand type should be considered as the main challenge. More importantly, these observation suggests that it is possible to obtain very strong bounds by considering only a small subset of demands, instead of the complete set of demands. In the sequel we further explore this direction.

6.2 Equivalent Bounds Using Subsets of Demands

As mentioned above, the investigation into single-demand-type caching systems suggests strong bounds can be obtained by considering systems with a small subset of demands. We hypothesize that in some cases, even equivalent bounds can be obtained, and moreover, these demands do not need to form a complete demand type class, and next we show that this is indeed the case. The two base cases we use are the $(N, K) = (2, 4)$ and $(N, K) = (3, 3)$ since for larger cases it appears complete computation even for a single-demand-type system may be too time-consuming or impossible.

To be more precise, we are relaxing the LP, by including only elemental inequality constraints that involve joint entropies of random variables within a subset of the random variables $\mathcal{W} \cup \mathcal{Z} \cup \mathcal{X}$, and other constraints are simply removed. However the symmetry structure specified in Section 3 is still maintained to reduce the problem. It should be noted that this is not equivalent to forming the LP on a caching system where only those files, users and demands are present, since in this alternative setting, symmetric solutions may induce loss of optimality.

There are many choices of subsets with which the outer bounds can be computed, and we only provide a few that are more relevant which confirm our conjecture:

Fact 6.4. *In terms of the computed outer bounds, the following facts are observed:*

- For the $(N, K) = (2, 4)$ case, the outer bound in Proposition 4.9 can be obtained by restricting to the subset of random variables $\mathcal{W} \cup \mathcal{Z} \cup \{X_{1,1,1,2}, X_{1,1,2,2}\}$.

- For the $(N, K) = (2, 4)$ case, the outer bound in Proposition 6.1 in the range $M \in [1/3, 2]$ for single demand type $(3, 1)$ can be obtained by restricting to the subset of random variables $\mathcal{W} \cup \mathcal{Z} \cup \{X_{2,1,1,1}, X_{1,2,1,1}, X_{1,1,2,1}, X_{1,1,1,2}\}$.
- For the $(N, K) = (3, 3)$ case, the intersection of the outer bounds in Proposition 6.2 can be obtained by restricting to the subset of random variables $\mathcal{W} \cup \mathcal{Z} \cup \{X_{2,1,1}, X_{3,1,1}, X_{3,2,1}\}$.
- For the $(N, K) = (3, 3)$ case, the outer bound in Proposition 6.2 in the range $M \in [2/3, 3]$ for single demand type $(2, 1)$ can be obtained by restricting to the subset of random variables $\mathcal{W} \cup \mathcal{Z} \cup \{X_{2,1,1}, X_{3,1,1}\}$.

These observations reveal that the subset of demands can be chosen rather small to produce strong bounds, for example, for the $(N, K) = (2, 4)$ case, including only 8 random variables produce the strongest bound as including all 22 random variables. Moreover, for specific regimes, the same bound can be produced using an even smaller number of random variables (for the case $(N, K) = (3, 3)$), or with a more specific set of random variables (for the case $(N, K) = (2, 4)$ where in the range $[1/3, 2]$, including only some of the demand type $(3, 1)$ is sufficient). Equipped with these insights, we can attempt to tackle larger problem cases, which would have appeared impossible to computationally produce meaningful outer bounds for. In the sequel, this approach is applied for two purposes: (1) to identify generic structures in converse proofs, and (2) to produce outer bounds for large problem cases.

6.3 Identifying Generic Structures in Converse Proofs

Recall our comment given after the proof of Theorem 4.12 that finding this proof is not straightforward. One critical clue was obtained when applying the exploration approach discussed above. When restricting the set of included random variables to a smaller set, the overall problem is being relaxed, however, if the outer bound thus obtained remains the same, it implies that the sought-after outer bound proof only needs to rely on the joint entropies within this restricted set. For the specific case of $(N, K) = (2, 5)$, we have the following fact.

Fact 6.5. *For $(N, K) = (2, 5)$, the bound $15M + 20R \geq 28$ in the range $M \in [6/5, 8/5]$ can be obtained by restricting to the subset of random variables $\mathcal{W} \cup \mathcal{Z} \cup \{X_{2,1,1,1,1}, X_{1,2,1,1,1}, X_{1,1,2,1,1}, X_{1,1,1,2,1}, X_{1,1,1,1,2}\}$.*

Together with the second item in Fact 6.4, we can naturally conjecture that in order to prove the hypothesized outer bound in Hypothesis 4.11, only the dependence structure within the set of random variables $\mathcal{W} \cup \mathcal{Z} \cup X_{\rightarrow[1:K]}$ needs to be considered, and all the proof steps can be written using mutual information or joint entropies of them alone. Although this is still not a trivial task, the possibility is significantly reduced: for the example of $(N, K) = (2, 5)$, to only 12 random variables, with a much simpler structure than that of the original problem with 39 random variables. Perhaps more importantly, such a restriction makes it feasible to identify common route of derivation in the converse proof and then generalize it, from which we obtain the proof of Theorem 4.12.

6.4 Computing Bounds for Larger Problem Cases

We now present a few outer bounds for larger problem cases, and make comparison with other known bounds in the literature. This is not intended to be a complete list of results we obtain, and more results will be made online while they are computed.

In Fig. 6, we provide results for $(N, K) = (4, 3)$, $(N, K) = (5, 3)$ and $(N, K) = (6, 3)$. Included are the computed outer bounds, the inner bound by the scheme in [1], the cut-set outer bounds, and for reference the outer bounds given in [5]. We omit the bounds in [6] and [7] to avoid too much clutter in the plot, however they do not provide better bounds than that in [5] for these cases. It can be seen that the computed bounds are in fact tight in the range $M \in [4/3, 4]$ for $(N, K) = (4, 3)$, $M \in [5/3, 5]$

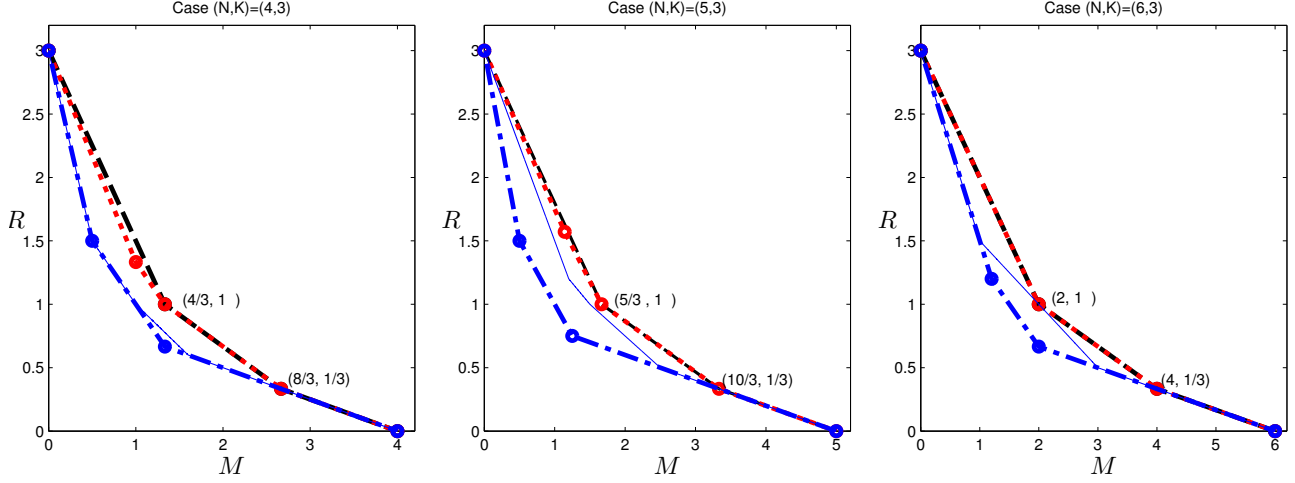


Figure 6: The computed outer bounds for $(N, K) = (4, 3)$, $(N, K) = (5, 3)$ and $(N, K) = (6, 3)$ caching systems. The red dotted lines give the computed outer bounds, the blue dashed-dot lines are the cut-set outer bounds, the black dashed lines are the inner bound using the scheme in [1], and the thin blue lines are the outer bounds given in [5]. Only nontrivial outer bound corner points that match inner bounds are explicitly labeled.

for $(N, K) = (5, 3)$, and tight in general for $(N, K) = (6, 3)$; in these ranges, the scheme given in [1] is in fact optimal. Unlike our computed bounds, the outer bound in [5] does not provide additional tight results beyond those already determined using the cut-set bound, except the single point $(M, R) = (2, 1)$ for $(N, K) = (6, 3)$.

In Fig. 7, we provide results for $(N, K) = (3, 4)$, $(N, K) = (3, 5)$ and $(N, K) = (3, 6)$. Included are the computed outer bounds, the inner bound by the scheme in [1] and that in [12], the cut-set outer bound, and for reference the outer bounds given in [5]. The bounds in [6] and [7] are again omitted. It can be seen that the computed bounds are in fact tight in the range $M \in [0, 1/4] \cup [3/2, 3]$ for $(N, K) = (3, 4)$, $M \in [0, 1/5] \cup [6/5, 3]$ for $(N, K) = (3, 5)$, and $M \in [0, 1/6] \cup [3/2, 3]$ for $(N, K) = (3, 6)$. Generally, in the high memory regime, the scheme given in [1] is in fact optimal, and in the low memory regime, the schemes in [8, 12] are optimal. It can be seen again that the outer bound in [5] does not provide additional tight results beyond those already determined using the cut-set bound.

The bounds given above (and other more computed results) in fact provide grounds and directions for further investigation and hypotheses on the optimal memory-transmission-rate tradeoff; some additional analytical results based on this data will be presented in a forthcoming work, in order to prevent distraction from the focus of the current paper, which is to provide several computer-aided investigation methods we have developed.

7 Conclusion

We presented a computer-aided investigation on the fundamental limit of the caching problem, including data-driven hypothesis forming which leads to several complete or partial characterizations of the memory-transmission-rate tradeoff, a new code construction reverse-engineered through the computed outer bounding data, and a computerized exploration approach that can reveal hidden structures in the problem and also enables us to find surprisingly strong outer bounds for larger problem cases.

It is our belief that this work provides strong evidence of the effectiveness of the computer-aided approach in the investigation of the fundamental limits of communication, data storage and data management systems. Although at the first sight, the exponential growth the LP problem would prevent any possibility of obtaining meaningful results on engineering problems of interest, our experience in [15] [28]

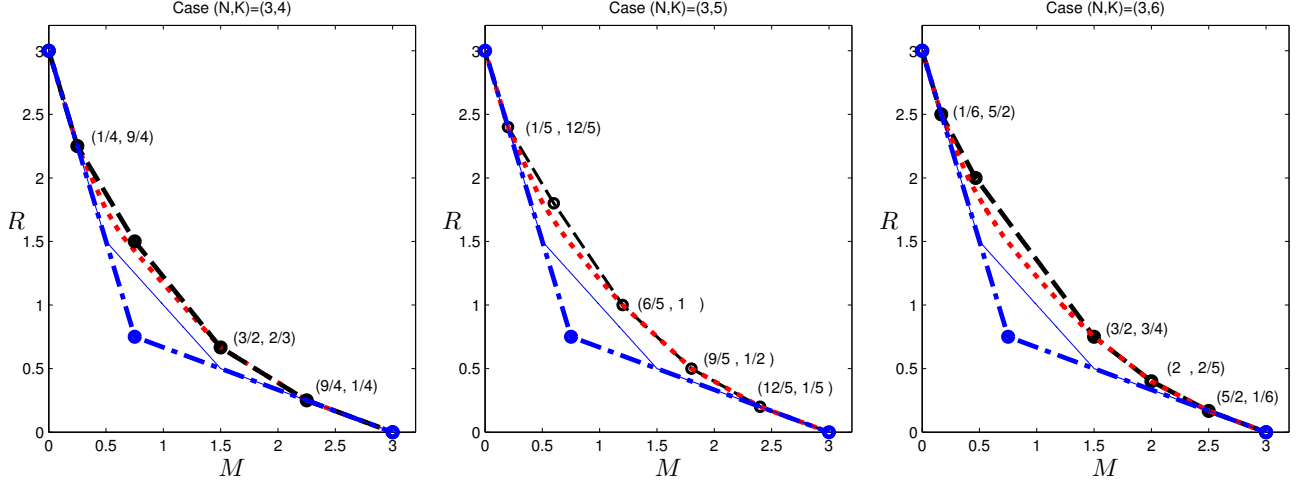


Figure 7: The computed outer bounds for $(N, K) = (3, 4)$, $(N, K) = (3, 5)$ and $(N, K) = (3, 6)$ caching systems. The red dotted lines give the computed outer bounds, the blue dashed-dot lines are the cut-set outer bounds, the black dashed lines are the inner bound using the scheme in [1] and [12], and the thin blue lines are the outer bounds given in [5]. Only nontrivial outer bound corner points that match inner bounds are explicitly labeled.

and the current work proves otherwise. By incorporating the structure of the problem, we develop more domain-specific tools in such investigations, and were able to obtain results that appear extremely difficult for human experts to obtain directly. This work in fact provides more insights into the caching problems, which will be explored and presented in a forthcoming paper. It is our hope that this work can serve as a starting point to introduce more machine intelligence and the corresponding computer-aided tools into information theory and communication research in the future.

Appendix A Finding Corner Points of the LP Outer Bounds

Since this is an LP problem, and also due to the problem setting, only the lower hull of the outer bound region between the two quantities M and R is of interest. The general algorithm in [16] is equivalent to the procedure given in Algorithm 1 in this specific setting. In this algorithm, the set \mathcal{P} in the input is the initial extreme points of the tradeoff region, which are trivially known from the problem setting. The variables and constraints in the LP are given as outlined in Section 2 for a fixed (N, K) pair, which are populated and considered fixed. The output set \mathcal{P} is the final computed extreme points of the outer bound. The algorithm can be intuitively explained as follows: starting with two known extreme points, if there are any other corner points, they must lie below the line segment connecting these two points, and thus an LP that minimizes the bounding plane along the direction of this line segment must be able to find a lower value; if so, the new point is also an extreme point and we can repeat this procedure again.

In the caching problem, the tradeoff is between two quantities M and R . We note here if there are more than two quantities which need to be considered in the tradeoff, the algorithm is more involved, and we refer the readers to [16] and [17] for more details on such settings.

Appendix B Tabulation Proofs of Proposition 4.1 and Proposition 4.2

The proof of the Proposition 4.1 is given in the Table 5-6, and that of the Proposition 4.2 is given in the Table 7-8. Each row in Table 6 and Table 8, except the last rows, are simple and known information inequalities, up to the symmetry defined in Section 3. The last rows in Table 6 and Table 8 are the sum

Algorithm 1: An algorithm to identify the corner points of the LP outer bound

Input : $N, K, \mathcal{P} = \{(N, 0), (0, \min(N, K))\}$

Output: \mathcal{P}

```

1  $n = 2; i = 1;$ 
2 while  $i < n$  do
3   Compute the line segment connecting  $i$ -th and  $(i + 1)$ -th  $(M, R)$  pairs in  $\mathcal{P}$ , as  $M + \alpha R = \beta$ ;
4   Set the objective of the LP as  $M + \alpha R$ , and solve LP for solution  $(M^*, R^*)$  and objective  $\beta^*$ ;
5   if  $\beta^* < \beta$  then
6     Insert  $(M^*, R^*)$  in  $\mathcal{P}$  between the  $i$ -th and  $(i + 1)$ -th  $(M, R)$  pairs;
7      $n = n + 1;$ 
8   else
9      $i = i + 1;$ 
10  end
11 end

```

of all previous rows, which are the sought-after inequalities.

Appendix C Proofs of Lemma 4.6 and Theorem 4.5

Proofs of Lemma 4.6. We first write the following chain of inequalities

$$\begin{aligned}
(N - n)H(Z_1, W_{[1:n]}, X_{n,n+1}) &= (N - n) [H(Z_1, W_{[1:n]}) + H(X_{n,n+1}|Z_1, W_{[1:n]})] \\
&\stackrel{(a)}{=} (N - n)H(Z_1, W_{[1:n]}) + \sum_{i=n+1}^N H(X_{n,i}|Z_1, W_{[1:n]}) \\
&\geq (N - n)H(Z_1, W_{[1:n]}) + H(X_{n,[n+1:N]}|Z_1, W_{[1:n]}) \\
&= (N - n - 1)H(Z_1, W_{[1:n]}) + H(X_{n,[n+1:N]}, Z_1, W_{[1:n]}), \tag{52}
\end{aligned}$$

where (a) is because of the file-index-symmetry. Next notice that by the user-index-symmetry

$$H(Z_1, W_{[1:n]}) = H(Z_2, W_{[1:n]}), \tag{53}$$

which implies that

$$\begin{aligned}
H(Z_1, W_{[1:n]}) + H(X_{n,[n+1:N]}, Z_1, W_{[1:n]}) &\geq H(Z_2, W_{[1:n]}) + H(X_{n,[n+1:N]}, W_{[1:n]}) \\
&\stackrel{(b)}{\geq} H(W_{[1:n]}) + H(X_{n,[n+1:N]}, Z_2, W_{[1:n]}) \\
&\stackrel{(c)}{=} H(W_{[1:n]}) + H(X_{n,[n+1:N]}, Z_2, W_{[1:N]}) \\
&= H(W_{[1:n]}) + H(W_{[1:N]}) = N + n, \tag{54}
\end{aligned}$$

where (b) is by the sub-modularity of the entropy function, and (c) is because of (3). Now substituting (54) into (52) gives (24), which completes the proof. \square

We are now ready to prove Theorem 4.5.

Proof of Theorem 4.5. For $N \geq 3$, it can be verified that the three corner points of the given tradeoff region are

$$(0, 2), \quad \left(\frac{N}{2}, \frac{1}{2}\right), \quad (N, 0), \tag{55}$$

T_1	F
T_2	$H(X_{1,2})$
T_3	$H(W_1)$
T_4	$H(W_3, X_{1,2})$
T_5	$H(W_1, W_2)$
T_6	$H(W_3, W_4, X_{1,2})$
T_7	$H(W_1, W_2, W_3, W_4)$
T_8	$H(Z_1)$
T_9	$H(Z_1, X_{1,2})$
T_{10}	$H(W_1, Z_1)$
T_{11}	$H(W_3, Z_1, X_{1,2})$
T_{12}	$H(W_1, W_2, Z_1)$
T_{13}	$H(W_3, W_4, Z_1, X_{1,2})$
T_{14}	$H(Z_1, Z_2, X_{1,2})$
T_{15}	$H(Z_1, Z_2, X_{1,2}, X_{1,3})$
T_{16}	$H(W_1, Z_1, Z_2)$
T_{17}	$H(W_3, Z_1, Z_2, X_{1,2})$
T_{18}	$H(W_1, W_2, Z_1, Z_2)$

Table 5: Terms needed to prove Proposition 4.1.

T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9	T_{10}	T_{11}	T_{12}	T_{13}	T_{14}	T_{15}	T_{16}	T_{17}	T_{18}
														1		-1	
	2	2	-2														
	4						4	-4									
						-1						2	-1	-1		2	-1
	-2							4				-2					
		-1	1	1	-1												
		-1	1						1	-1							
				-1	1						1	-1					
									-1	1					1	-1	
						-1					-1	1					1
-8						2											
-8	4						4										

Table 6: Proof by Tabulation of Proposition 4.1, with terms defined in Table 5.

T_1	F
T_2	$H(X_{1,2})$
T_3	$H(W_1)$
T_4	$H(W_1, X_{1,2})$
T_5	$H(W_3, X_{1,2})$
T_6	$H(W_1, X_{1,2}, X_{1,3})$
T_7	$H(W_1, X_{1,2}, X_{1,3}, X_{1,4})$
T_8	$H(W_1, W_2)$
T_9	$H(W_1, W_3, X_{1,2})$
T_{10}	$H(W_1, W_4, X_{1,2}, X_{1,3})$
T_{11}	$H(W_1, W_2, W_3, W_4)$
T_{12}	$H(Z_1)$
T_{13}	$H(Z_1, X_{1,2})$
T_{14}	$H(Z_1, X_{1,2}, X_{1,3})$
T_{15}	$H(Z_1, X_{1,2}, X_{1,3}, X_{1,4})$
T_{16}	$H(W_1, Z_1)$
T_{17}	$H(W_3, Z_1, X_{1,2})$
T_{18}	$H(W_2, Z_1, X_{1,2})$
T_{19}	$H(W_4, Z_1, X_{1,2}, X_{1,3})$
T_{20}	$H(W_1, Z_2, X_{1,2}, X_{1,3})$
T_{21}	$H(W_1, W_2, Z_1)$
T_{22}	$H(W_2, W_3, Z_1, X_{1,2})$

Table 7: Terms needed to prove Proposition 4.2.

T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9	T_{10}	T_{11}	T_{12}	T_{13}	T_{14}	T_{15}	T_{16}	T_{17}	T_{18}	T_{19}	T_{20}	T_{21}	T_{22}
									-1									1			
						-2								2							
	2	2		-2																	
	6										6	-6									
											8	-4		-4							
																2		-1		-1	
											-2	4	-2								
		-2	2												2		-2				
		-2		2											2	-2					
							-1	1												1	-1
			-2		2												2		-2		
							-1	1	-1												1
					-2	2				-2									2		
-2		2																			
-2							1														
-12										3											
-16	8												6								

Table 8: Proof by Tabulation of Proposition 4.2, with terms defined in Table 7.

which are achievable using the codes given in [1]. The outer bound $M + NR \geq N$ can also be obtained as one of the cut-set outer bounds in [1], and it only remains to show that the inequality $3M + NR \geq 2N$ is true. For this purpose, we claim that for any integer $n \in \{1, 2, \dots, N-2\}$

$$\begin{aligned}
3M + NR &\geq 3 \sum_{j=1}^n \left[\frac{N+j}{N-j} \prod_{i=1}^{j-1} \frac{N-(i+2)}{N-i} \right] \\
&\quad + 3 \prod_{j=1}^n \frac{N-(j+2)}{N-j} H(Z_1, W_{[1:n]}) \\
&\quad + [N-(n+2)] \prod_{j=1}^{n-1} \frac{N-(j+2)}{N-j} H(X_{1,2}),
\end{aligned} \tag{56}$$

which we prove next by induction.

First notice that

$$\begin{aligned}
3M + NR &\geq 3H(Z_1) + NH(X_{1,2}) \\
&\geq 3H(Z_1, X_{1,2}) + (N-3)H(X_{1,2}) \\
&\stackrel{(3)}{=} 3H(Z_1, W_1, X_{1,2}) + (N-3)H(X_{1,2}) \\
&\stackrel{(d)}{\geq} \frac{3(N-3)}{N-1} H(Z_1, W_1) + \frac{3(N+1)}{N-1} + (N-3)H(X_{1,2}),
\end{aligned} \tag{57}$$

where we wrote (3) to mean by Eqn. (3), and (d) is by Lemma 4.6 with $n = 1$. This is precisely the claim when $n = 1$, when we take the convention $\prod_k^n(\cdot) = 1$ when $n < k$ in (56).

Assume the claim is true for $n = n^*$, and we next prove it is true for $n = n^* + 1$. Notice that the second and third terms in (56) has a common factor

$$\frac{N-(n^*+2)}{N-n^*} \prod_{j=1}^{n^*-1} \frac{N-(j+2)}{N-j} = \prod_{j=1}^{n^*} \frac{N-(j+2)}{N-j}, \tag{58}$$

using which to normalize the last two terms gives

$$\begin{aligned}
&3H(Z_1, W_{[1:n^*]}) + (N-n^*)H(X_{1,2}) \\
&\stackrel{(e)}{=} 3[H(Z_1, W_{[1:n^*]}) + H(X_{n^*+1, n^*+2})] \\
&\quad + (N-n^*-3)H(X_{1,2}) \\
&\geq 3[H(Z_1, W_{[1:n^*]}, X_{n^*+1, n^*+2})] \\
&\quad + (N-n^*-3)H(X_{1,2}) \\
&\stackrel{(3)}{=} 3[H(Z_1, W_{[1:n^*+1]}, X_{n^*+1, n^*+2})] \\
&\quad + (N-n^*-3)H(X_{1,2}) \\
&\stackrel{(f)}{\geq} 3 \frac{(N-n^*-3)}{N-n^*-1} H(Z_1, W_{[1:n^*+1]}) + 3 \frac{N+n^*+1}{N-n^*-1} \\
&\quad + (N-n^*-3)H(X_{1,2}),
\end{aligned} \tag{59}$$

where (e) is by the file-index-symmetry, and (f) is by Lemma 4.6. Substituting (58) and (59) into (56) for the case $n = n^*$ gives exactly (56) for the case $n = n^* + 1$, which completes the proof for (56).

It remains to show that (56) implies the bound $3M + NR \geq 2N$. For this purpose, notice that when $n = N - 2$, the last two terms in (56) reduce to zero, and thus we only need to show that

$$Q(N) \triangleq 3 \sum_{j=1}^{N-2} \left[\frac{N+j}{N-j} \prod_{i=1}^{j-1} \frac{N-(i+2)}{N-i} \right] = 2N. \quad (60)$$

For each summand, we have

$$\begin{aligned} \frac{N+j}{N-j} \prod_{i=1}^{j-1} \frac{N-(i+2)}{N-i} &= \frac{N+j}{N-j} \left[\frac{N-3}{N-1} \frac{N-4}{N-2} \frac{N-5}{N-3} \cdots \frac{N-j-1}{N-j+1} \right] \\ &= \frac{(N-j-1)(N+j)}{(N-1)(N-2)}. \end{aligned} \quad (61)$$

Thus we have

$$Q(N) = \frac{3}{(N-1)(N-2)} \sum_{j=1}^{N-2} (N-j-1)(N+j) = 2N,$$

where we have used the well-known formula for the sum of integer squares. The proof is thus complete. \square

Appendix D Proof of Proposition 4.8

We first consider the achievability, for which only the achievability of the following extremal points needs to be shown because of the polytope structure of the region:

$$(M, R) \in \left\{ (0, 2), \left(\frac{1}{3}, \frac{4}{3} \right), \left(\frac{4}{3}, \frac{1}{3} \right), (2, 0) \right\}. \quad (62)$$

Achieving the rate pairs $(0, 2)$ and $(2, 0)$ is trivial. The scheme in [1] can achieve the rate pair $(\frac{4}{3}, \frac{1}{3})$. The rate pair $(\frac{1}{3}, \frac{4}{3})$ can be achieved by a scheme given in [8], which is a generalization of a special scheme given in [1].

To prove the converse, we note first that the cut-set-based approach can provide all bounds in (25) except

$$3M + 3R \geq 5, \quad (63)$$

which is a new inequality. As mentioned earlier, this inequality is a special case of Theorem 4.12 and there is no need to prove it separately.

Appendix E Proof of Lemma 4.13

Proof of Lemma 4.13. We prove this lemma by induction. First consider the case when $k = K - 1$, for which we write

$$\begin{aligned} &2H(Z_1, W_1, X_{\rightarrow[2:K-1]}) \\ &\stackrel{(a)}{=} H(Z_1, W_1, X_{\rightarrow[2:K-1]}) + H(Z_1, W_1, X_{\rightarrow[2:K-2]}, X_{\rightarrow K}) \\ &= H(X_{\rightarrow K-1} | Z_1, W_1, X_{\rightarrow[2:K-2]}) + H(X_{\rightarrow K} | Z_1, W_1, X_{\rightarrow[2:K-2]}) + 2H(Z_1, W_1, X_{\rightarrow[2:K-2]}) \\ &\geq H(Z_1, W_1, X_{\rightarrow[2:K]}) + H(Z_1, W_1, X_{\rightarrow[2:K-2]}), \end{aligned} \quad (64)$$

where (a) is by file-index symmetry. The first quantity can be lower bounded as

$$H(Z_1, W_1, X_{\rightarrow[2:K]}) \geq H(W_1, X_{\rightarrow[2:K]}), \quad (65)$$

which leads to a bound on the following sum

$$\begin{aligned} & H(Z_1, W_1, X_{\rightarrow[2:K]}) + H(Z_1, W_1, X_{\rightarrow[2:K-1]}) \\ & \geq H(W_1, X_{\rightarrow[2:K]}) + H(Z_1, W_1, X_{\rightarrow[2:K-1]}) \\ & \geq H(X_{\rightarrow K}|W_1, X_{\rightarrow[2:K-1]}) + H(Z_1|W_1, X_{\rightarrow[2:K-1]}) + 2H(W_1, X_{\rightarrow[2:K-1]}) \\ & \stackrel{(b)}{=} H(X_{\rightarrow K}|W_1, X_{\rightarrow[2:K-1]}) + H(Z_K|W_1, X_{\rightarrow[2:K-1]}) + 2H(W_1, X_{\rightarrow[2:K-1]}) \\ & \geq H(Z_K, X_{\rightarrow K}|W_1, X_{\rightarrow[2:K-1]}) + 2H(W_1, X_{\rightarrow[2:K-1]}) \\ & \stackrel{(c)}{=} H(Z_K, X_{\rightarrow K}, W_2|W_1, X_{\rightarrow[2:K-1]}) + 2H(W_1, X_{\rightarrow[2:K-1]}) \\ & \stackrel{(d)}{=} H(W_1, W_2) + H(W_1, X_{\rightarrow[2:K-1]}), \end{aligned} \quad (66)$$

where (b) is by the user index symmetry, and (c) is because Z_K and $X_{1,1}, \dots, X_{1,2}$ can be used to produce W_2 , and (d) is because all other variables are deterministic functions of (W_1, W_2) . Adding $H(Z_1, W_1, X_{\rightarrow[2:K-1]})$ on both sides of (64), and then apply (66) leads to

$$\begin{aligned} 3H(Z_1, W_1, X_{\rightarrow[2:K-1]}) & \geq H(Z_1, W_1, X_{\rightarrow[2:K-2]}) + H(W_1, X_{\rightarrow[2:K-1]}) + H(W_1, W_2) \\ & \stackrel{(e)}{=} H(Z_{K-1}, W_1, X_{\rightarrow[2:K-2]}) + H(W_1, X_{\rightarrow[2:K-1]}) + H(W_1, W_2) \\ & \stackrel{(f)}{\geq} H(Z_{K-1}, W_1, X_{\rightarrow[2:K-1]}) + H(W_1, X_{\rightarrow[2:K-2]}) + H(W_1, W_2) \\ & = H(Z_{K-1}, W_1, X_{\rightarrow[2:K-1]}, W_2) + H(W_1, X_{\rightarrow[2:K-2]}) + H(W_1, W_2) \\ & = H(W_1, X_{\rightarrow[2:K-2]}) + 2H(W_1, W_2), \end{aligned} \quad (67)$$

which (e) follows from the user-index symmetry, and (f) by the sub-modularity of the entropy function. This is precisely (31) for $k = K - 1$.

Now suppose (31) holds for $k = k^* + 1$, we next prove it is true for $k = k^*$ for $K \geq 4$, since when $K = 3$ there is nothing to prove beyond $k = K - 1 = 2$. Using a similar decomposition as in (64), we can write

$$2H(Z_1, W_1, X_{\rightarrow[2:k^*]}) \geq H(Z_1, W_1, X_{\rightarrow[2:k^*+1]}) + H(Z_1, W_1, X_{\rightarrow[2:k^*-1]}) \quad (68)$$

Next we apply the supposition for $k = k^* + 1$ on the first term of the right hand side, which gives

$$\begin{aligned} 2H(Z_1, W_1, X_{\rightarrow[2:k^*]}) & \geq \frac{[(K - k^* - 1)(K - k^*) - 2]H(Z_1, W_1, X_{\rightarrow[2:k^*]})}{(K - k^*)(K - k^* + 1)} + \frac{2H(W_1, X_{\rightarrow[2:k^*]})}{(K - k^*)(K - k^* + 1)} \\ & \quad + \frac{2(K - k^*)H(W_1, W_2)}{(K - k^*)(K - k^* + 1)} + H(Z_1, W_1, X_{\rightarrow[2:k^*-1]}) \end{aligned} \quad (69)$$

Notice that the coefficient in front of $H(W_1, X_{\rightarrow[2:k^*]})$ is always less than one for $K \geq 4$ and $k^* \in \{2, 3, \dots, K - 1\}$, and we can thus bound the following sum

$$\begin{aligned} & \frac{2H(W_1, X_{\rightarrow[2:k^*]})}{(K - k^*)(K - k^* + 1)} + H(Z_1, W_1, X_{\rightarrow[2:k^*-1]}) \\ & = \frac{2[H(W_1, X_{\rightarrow[2:k^*]}) + H(Z_1, W_1, X_{\rightarrow[2:k^*-1]})]}{(K - k^*)(K - k^* + 1)} + \frac{(K - k^*)(K - k^* + 1) - 2}{(K - k^*)(K - k^* + 1)} H(Z_1, W_1, X_{\rightarrow[2:k^*-1]}) \\ & \stackrel{(g)}{\geq} \frac{2[H(W_1, W_2) + H(W_1, X_{\rightarrow[2:k^*-1]})]}{(K - k^*)(K - k^* + 1)} + \frac{(K - k^*)(K - k^* + 1) - 2}{(K - k^*)(K - k^* + 1)} H(Z_1, W_1, X_{\rightarrow[2:k^*-1]}), \end{aligned} \quad (70)$$

where (g) follows the same line of argument as in (66). Substituting (70) into (69) and canceling out the common terms of $H(Z_1, W_1, X_{\rightarrow[2:k^*]})$ on both sides now give (31) for $k = k^*$. The proof is thus complete. \square

Appendix F Proof for the Forward of Proposition 6.1

Note that the optimal tradeoff for the single demand type $(3, 1)$ system has the following corner points

$$(M, R) = (0, 2), \left(\frac{1}{4}, \frac{3}{2}\right), \left(\frac{1}{2}, \frac{7}{6}\right), \left(1, \frac{2}{3}\right), \left(\frac{3}{2}, \frac{1}{4}\right), (2, 0).$$

The corner points $(1, \frac{2}{3})$ and $(\frac{3}{2}, \frac{1}{4})$ are achievable using the Maddah-Ali-Niesen scheme [1]. The point $(\frac{1}{4}, \frac{3}{2})$ is achievable by the code given in [8] or [12]. The only remaining corner point of interest is thus $(\frac{1}{2}, \frac{7}{6})$, in the binary field. This can be achieved by the following strategy in Table 9, where the first file has 6 symbols (A_1, A_2, \dots, A_6) and the second file (B_1, B_2, \dots, B_6) . By the symmetry, we only need to

User 1	$A_1 + B_1$	$A_2 + B_2$	$A_3 + B_3$
User 2	$A_1 + B_1$	$A_4 + B_4$	$A_5 + B_5$
User 3	$A_2 + B_2$	$A_4 + B_4$	$A_6 + B_6$
User 4	$A_3 + B_3$	$A_5 + B_5$	$A_6 + B_6$

Table 9: Code for the tradeoff point $(\frac{1}{2}, \frac{7}{6})$ for demand type $(3, 1)$ when $(N, K) = (2, 4)$.

consider the demand when the first three users request A and the last user request B . The server can send the following symbols in this case

$$A_3, A_5, A_6, B_1, B_2, B_4, A_1 + A_2 + A_4.$$

Let us consider now the single demand type $(2, 2)$ system, for which the corner points on the optimal tradeoff are:

$$(M, R) = (0, 2), \left(\frac{1}{3}, \frac{4}{3}\right), \left(\frac{4}{3}, \frac{1}{3}\right), (2, 0).$$

Let us denote the first file as (A_1, A_2, A_3) , and the second file as (B_1, B_2, B_3) , which are in the binary field. To achieve the corner point $(\frac{1}{3}, \frac{4}{3})$, we use the caching code in Table 10. Again due to the symmetry,

User 1	$A_1 + B_1$
User 2	$A_2 + B_2$
User 3	$A_3 + B_3$
User 4	$A_1 + A_2 + A_3 + B_1 + B_2 + B_3$

Table 10: Code for the tradeoff point $(\frac{1}{3}, \frac{4}{3})$ for demand type $(2, 2)$ when $(N, K) = (2, 4)$.

we only need to consider the case when the first two users request A , and the other two request B . For this case, the server can send

$$B_1, B_2, A_3, A_1 + A_2 + A_3.$$

For the other corner point $(\frac{4}{3}, \frac{1}{3})$ the following placement in Table 11 can be used. Again for the case when the first two users request A , and the other two request B , the server can send

$$A_1 - A_3 + B_2.$$

User 1	A_1	A_2	B_1	B_2
User 2	A_2	A_3	B_2	B_3
User 3	A_1	A_3	B_1	B_3
User 4	$A_1 + A_2$	$A_2 + A_3$	$B_1 + B_2$	$B_2 + B_3$

Table 11: Code for the tradeoff point $(\frac{4}{3}, \frac{1}{3})$ for demand type $(2, 2)$ when $(N, K) = (2, 4)$.

Acknowledgment

The author wishes to thank Dr. Urs Niesen and Dr. Vaneet Aggarwal for early discussions which partly motivated this work. He also wishes to thank Dr. Jun Chen for several discussions as well as the insightful comments on segments of a draft. Additionally the author wishes to thank the authors of [5] for making the source code to compute several outer bounds available online, which was conveniently used for several comparisons in this work.

References

- [1] M. A. Maddah-Ali and U. Niesen, “Fundamental limits of caching,” *IEEE Trans. on Information Theory*, vol. 60, no. 5, pp. 2856–2867, May 2014.
- [2] —, “Decentralized coded caching attains order-optimal memory-rate tradeoff,” *IEEE/ACM Transactions on Networking*, vol. 23, no. 4, pp. 1029–1040, Aug. 2015.
- [3] —, “Coded caching with nonuniform demands,” in *Proc. of INFOCOM Workshops 2014*, Toronto, Canada, Apr.-May 2014, pp. 221–226.
- [4] R. Pedarsani, M. A. Maddah-Ali, and U. Niesen, “Online coded caching,” in *Proc. of 2014 IEEE International Conference on Communications (ICC)*, Sydney, Australia, Jun. 2014, pp. 1878–1883.
- [5] H. Ghasemi and A. Ramamoorthy, “Improved lower bounds for coded caching,” in *Proc. of 2015 IEEE International Symposium on Information Theory (ISIT)*, Hong Kong, China, Jun. 2015.
- [6] A. Sengupta, R. Tandon, and T. C. Clancy, “Improved approximation of storage-rate tradeoff for caching via new outer bounds,” in *Proc. of 2015 IEEE International Symposium on Information Theory (ISIT)*, Hong Kong, China, Jun. 2015.
- [7] Ajaykrishnan N., N. S. Prem, V. M. Prabhakaran, and R. Vaze, “Critical database size for effective caching,” *arXiv:1501.02549*.
- [8] Z. Chen, P. Fan, and K. B. Letaief, “Fundamental limits of caching: Improved bounds for small buffer users,” *arXiv:1407.1935*, 2014.
- [9] S. Sahraei and M. Gastpar, “ k users caching two files: An improved achievable rate,” *arXiv:1512.06682*, 2015.
- [10] M. Amiri, Q. Yang, and D. Gunduz, “Coded caching for a large number of users,” *arXiv:1605.01993*, 2016.
- [11] K. Wan, D. Tuninetti, and P. Piantanida, “On caching with more users than files,” *arXiv:1601.06383*.
- [12] C. Tian and J. Chen, “Caching and delivery via interference elimination,” in *Proc. of 2016 IEEE International Symposium on Information Theory (ISIT)*, Barcelona, Spain, Jul. 2016.

- [13] R. W. Yeung, “A framework for linear information inequalities,” *IEEE Trans. on Information Theory*, vol. 43, no. 6, pp. 1924–1934, Nov. 1997.
- [14] R. Yeung, *A First Course in Information Theory*. New York: Kluwer Academic Publishers, 2002.
- [15] C. Tian, “Characterizing the rate region of the $(4, 3, 3)$ exact-repair regenerating codes,” *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 5, pp. 967–975, May 2014.
- [16] C. Lassez and J.-L. Lassez, “Quantifier elimination for conjunctions of linear constraints via a convex hull algorithm,” in *Symbolic and numerical computation for artificial intelligence*, B. R. Donald, D. Kapur, and J. L. Mundy, Eds. San Diego, CA: Academic Press, 1992, ch. 4, pp. 103–1199.
- [17] J. Apte and J. M. Walsh, “Exploiting symmetry in computing polyhedral bounds on network coding rate regions,” in *Proc. of International Symposium on Network Coding (NetCod) 2015*, Sydney, Australia, Jun. 2015, pp. 76–80.
- [18] C. Li, S. Weber, and J. M. Walsh, “Multilevel diversity coding systems: Rate regions, codes, computation, & forbidden minors,” *arXiv:1407.5659*, 2014.
- [19] S.-W. Ho, C. W. Tan, and R. W. Yeung, “Proving and disproving information inequalities,” in *Proc. of 2014 IEEE International Symposium on Information Theory (ISIT)*, Honolulu, HI, Jun. 2014.
- [20] K. Zhang and C. Tian, “Symmetry reduction of information inequalities,” in *54th Annual Allerton Conference on Communication, Control and Computing*, Monticello, IL, Sep. 2016.
- [21] G. E. Andrews, *The Theory of Partitions*. Cambridge University Press, 1976.
- [22] C. Tian, “Solutions of computed information theoretic limits (SCITL),” <http://web.eecs.utk.edu/~ctian1/SCITL.html>.
- [23] R. Dougherty, C. Freiling, and K. Zeger, “Insufficiency of linear coding in network information flow,” *IEEE Trans. on Information Theory*, vol. 51, no. 8, pp. 2745–2759, Aug. 2005.
- [24] Z. Zhang and R. W. Yeung, “A non-shannon-type conditional inequality of information quantities,” *IEEE Trans. on Information Theory*, vol. 43, no. 6, pp. 1982–1986, Nov. 1997.
- [25] —, “On characterization of entropy function via information inequalities,” *IEEE Trans. on Information Theory*, vol. 44, no. 4, pp. 1440–1452, Jul. 1998.
- [26] R. W. Yeung and Z. Zhang, “On symmetrical multilevel diversity coding,” *IEEE Trans. on Information Theory*, vol. 45, no. 2, pp. 609–621, Mar. 1999.
- [27] C. Tian, “Latent capacity region: A case study on symmetric broadcast with common messages,” *IEEE Trans. on Information Theory*, vol. 57, no. 6, pp. 3273–3285, Jun. 2011.
- [28] C. Tian and T. Liu, “Multilevel diversity coding with regeneration,” *IEEE Trans. on Information Theory*, vol. 62, no. 9, pp. 4833–4847, Sep. 2016.